eserver

iSeries

Performance

eserver

iSeries

Performance

Contents

Performance

How much do you invest in managing the performance of your system? The needs of your business change, sometimes sooner than you expect. To respond to business changes effectively, your system must change too. Managing your system, at first glance, might seem like just another time-consuming job. But the investment pays off soon because the system runs more efficiently, and this is reflected in your business. It is efficient because changes are planned and managed.

Managing performance on an iSeries server can be a complex task that requires a thorough understanding of iSeries work management. Understanding all the different processes that affect system performance can be challenging for the inexperienced user. Resolving performance problems requires the effective use of a large suite of tools, each with their own unique set of requirements and supported functions. Even after you have gathered and analyzed performance data, knowing what to do with that information can be daunting.

This topic will guide you through the tasks and tools associated with performance management.

What's new for V5R2
This topic describes what information is new or significantly changed in this release.

Print this topic
If you prefer a printed version of this information, go here to print the PDF.

Plan for performance
Setting performance objectives for your iSeries server will allow you to have measurable performance benchmarks to compare your performance data. This topic will explain how to set those benchmarks and how to use them later.

Set up your environment to manage performance
iSeries servers include powerful applications for managing system performance. However, they must be properly configured in order to meet the specific needs of your unique business environment. Learn how to configure applications to routinely collect, monitor, and analyze performance data.

Manage performance
Performance management is necessary to optimize utilization of your computer system by measuring current capabilities, recognizing trends, and making appropriate adjustments to satisfy end user and management requirements such as response time or job throughput. It is needed to maintain business efficiency and avoid prolonged suspension of normal business activities. Therefore, managing performance is part of your daily operations.

Applications for performance management
Managing performance on iSeries systems requires the use of a variety of specialized applications. Each of these applications offers a specific insight into system performance. This topic explains several applications and the intended use of each application.

Scenarios: Performance
One of the best ways to learn about performance management is to see examples that illustrate how you can use these applications or tools in your business environment. Find out about these examples.

Related information
IBM$^{(R)}$ related information contains technical, know-how, and "how to" information.

Note: This topic contains code examples. Read the Code example disclaimer for important legal information.

What's new in V5R2

Here is what is new for this release:

- **iSeries Navigator monitors**
  Find out about the new monitor support and the changes to the existing monitors.
- **Collection Services**
  Read about the changes to the performance database files and about the new function.
- **Performance Tools**
  Read about the changes to the reports.
- **Performance Management/400**
  Read about how PM/400 automatically collects data.
- **Performance explorer**
  Read about the changes to the performance explorer database files and about the new function.
- **Capacity Upgrade on Demand**
  Read about iSeries support for Capacity Upgrade on Demand.
- **Manage e-business performance**
  Read about the tasks, considerations, and resources available to manage iSeries performance in an e-business environment.
- **Main storage affinity**
  Read about how processes and threads can achieve improved affinity for memory and processor resources.

What's new: Monitors

Here is what is new for this release:

- File monitor: A function that lets you manage a multiple-platform environment. You will be able to run commands, monitor files, and manage multi-platform endpoint systems in iSeries Navigator.
- B2B activity monitor: An activity monitor that allows you to view transaction data with detailed information on the specified consolidated system and to automatically run OS/400 commands to control the overall transaction count or the duration of a transaction.
- The system monitor metrics now include information associated with the Point-to-Point Protocol. Several metrics were added to the CPU utilization properties. These new metrics relate to file systems, stream files, journal operations, and counter sets.

What's new: Collection Services

Here is what is new for this release:

Installing a new release
You have several choices as to how to handle your performance data when upgrading to a new release. You can continue to convert the data in your collection library with the Convert Performance Data (CVTPFRDTA) command. You can use the new automatic data collection support. If you do not convert your performance database files, and if Collection Services is started and you specified the option to create database files automatically, a performance library for the prior release files is created. The performance database files are moved to that library. This action allows new files to be created and preserves the existing data files from the previous release level. Finally, if you do not convert your performance database files and do not run Collection Services, you can delete the QAPMxxxx files with the Delete File (DLTF) command (DLTF library/QAPM*).

User-defined categories
Prior to V5R2, all performance collection categories were predefined, and all data collection programs were shipped with the system. Beginning in V5R2, you can define your own performance collection categories. The user-defined performance collection categories allow you to:

- Add new performance collection categories to the collector. The Register Collector Data Category (QypsRegCollectorDataCategory) adds a user-defined data category to one or more collector definitions of the Collection Services function of Management Central.
- Provide a data collection program for the new category to collect the data
- Store data that is collected for the category in a management collection object
- Schedule and run data collection for the category in the same way as for system-defined categories
- Access data that is collected for the new category in a management collection object with the Management Collection Object APIs.

Domino and HTTP server (powered by Apache) utilize this new feature to integrate their performance data into Collection Services.

User-defined transactions
Beginning in V5R2, Collection Services takes advantage of the performance explorer transaction boundary APIs to give application users the ability to define their own transactions. The Collection Services data category is USRTNS, and the performance database file associated with user-defined transactions is QAPMUSRTNS. These APIs include the following:

- Start Transaction
- End Transaction

Performance database files
All the database files, with the exception of QAPMAPPN, are now interactive, which means that you can sort the text in the columns alphabetically and you can search for specific words in the table.

The following table shows the new and changed database files.

Database file	Description
QAPMJOBWT	New file that contains data for job, task, and thread wait conditions.
QAPMJOBWTD	New file that contains a description of the counter sets found in file QAPMJOBWT.
QAPMHTTPB	New file that contains the basic data for IBM HTTP Server (powered by Apache).
QAPMHTTPD	New file that contains detailed data for IBM HTTP Server (powered by Apache).
QAPMDOMINO	New file that contains data for Domino for iSeries.
QAPMPPP	New file that contains data for the Point-to-Point Protocol.
QAPMUSRTNS	New file that contains data for user-defined transactions.
QAPMSYSTEM	New fields that support file system counters and journal counters.
QAPMSYS	New field that provides a way to determine the difference between the system clocks on different partitions of a single system.
QAPMECL	New field that reports the number of discarded frames from unsupported protocols.
QAPMETH	New field that reports the number of discarded frames from unsupported protocols.
QAPMCONF	New record keys.
QAPMIOPD	New fields that support I/O adapter data.

Database file	Description
QAPMMIOP	New fields that support processing time.
QAPMJOBMI	New fields that support file system counters and journal counters.
QAPMDISK	New and changed fields that support independent ASP-related changes.
QAPMJOBOS	New fields that support file system counters.

What’s new: Performance Tools licensed program

Here is what is new for this release:

- BEST/1 is being withdrawn from the Performance Tools effective with this release. Alternatives that you may consider for sizing your capacity requirements include, but are not limited to, IBM’s Workload Estimator (integrated with PM/400) and BMC’s PATROL for iSeries (AS/400) - Predict product for distributed systems. Support for the following commands or menu options will be withdrawn from the Performance Tools licensed program effective with V5R2:
  - Start BEST/1 (Start BEST/1)
  - Analyze BEST/1 Model (ANZBESTMDL)
  - Create BEST/1 Model (CRTBESTMDL)
  - Convert MDLSYS File (CVTMDLSYSF)
  - Delete BEST/1 Model (DLTBESTMDL)
  - Print BEST/1 File (PRTBESTF)
  - Check Version of Specified Performance Data Member (QCYCHKV)
  - Convert System/36 Files from MDLSYS to BEST/1 (QCYCVTBD)
  - Find First and Last Date of Specified Performance Data Member (QCYFLDT)
  - Receive Messages from Previous Layer and Resend to Invoking Layer (QCYRSNDM)
  - Start MDLSYS (MDLSYS)
  - Capacity planning/modeling has been removed from the the PERFORM menu.
- The function associated with the Display Performance Data (DSPPFRDTA) command is now available from iSeries Navigator as the Performance Tools plugin. The interface provides function that is nearly equivalent to the DSPPFRDTA command. It allows you to generate and view a subset of the Performance Tools licensed program reports.
- The Print Component Report (PRTCPTRPT) command was updated to reflect several enhancements, which include the following:
  - A new selection category allows you to select the job types to print. This selection is helpful for analyzing interactive work on a server model.
  - A new selection category allows you to select which job priorities to include in your report. This selection is helpful for getting totals of the high priority work and finding all priority 00 work when a system has high overhead.
  - A new column reports the percentage of write cache overruns (% write cache overruns).
  - The Database Journaling Summary section includes a new subsection that shows information related to journal counters and operations. The data for the new counters are stored in the QAPMJOBMI file.
- The Print System Report (PRTSYSRPT) command was updated to reflect several enhancements, which include the following:
  - The Communications Summary now includes the Point-to-Point Protocol (PPP) information. The PPP protocol is also shown in the Communications Detail section of the Resource Report. The Display Performance Data (DSPPFRDTA) command also shows Point-to-Point Protocol information from the Display Communications Line Detail display.

  – A new section was added that shows information about transactions processed by HTTP Server jobs. This information is presented for each interval with a summary and average line at the bottom of the report.
  – New categories were included in the Workload, Resource Utilization, and Resource Utilization Expansion sections to show more detail about non-interactive server jobs.
- The Work with System Activity display shows the current processing capacity. This information is useful in shared processor environments, especially in cases where the capacity of the partition changes but the number of virtual processors does not change. The Work with System Activity display also shows the total percent of time spent by the job or task in a waiting state. You can view a list of the wait categories with option 6 (Wait detail). Several fields that relate to the wait time accounting data are added to the QAITMON file, which is created by the WRKSYSACT command.
- The Summary of Seize/Lock Conflicts by Object section of the Transaction Report can display a maximum of 500 objects. It is possible that trace data can have more than 500 seize/lock conflicts. Therefore, the 500 most significant objects will be shown in this section.
- The Start Performance Trace (STRPFRTRC) command now supports larger trace data tables.
- The size of some of the fields in the reports was increased to show greater values. The change affects the following reports.

Report	**Section**	**Field**
System	Resource Utilization	Disk I/O per Second
System	Resource Utilization Expansion	Physical Disk I/O, Logical Data Base I/O, Disk I/O (Sync and Async), DIO/Sec (Sync and Async)
Component	Component Interval Activity	Disk I/O per Second
Job Interval	Non-interactive Job Summary	Number of I/O per Second
Job Interval	Non-interactive Job Detail	Nbr I/O /Sec

What's new: Performance Management/400

Here is what is new for this release:

PM/400 automatically collects performance data, unless you upgrade from a previous release and turn off PM/400 performance data collection on the previous release. The data is not transmitted to IBM until you give permission to send it to IBM. The benefit of automatically collecting performance data is that you have the data when you need it and you can receive the PM/400 reports sooner after you activate PM/400.

For the most current information about the PM/400e reports, go to the PM/400 Web site

.

Converting performance data

Because performance database files are downlevel after you install a new release, you need to perform an action to allow the collecting of performance data to continue. If you use Performance Management/400, Collection Services, or the Performance Tools licensed program, and you collected data prior to installing your new release, you have the following choices regarding how to handle your performance data from the prior release:

- Convert the data
  Use the Convert Performance Data (CVTPFRDTA) command to convert the data in the collection library.
-

- Use the automatic data collection support
  If you do not convert your performance database files, and if Collection Services is started and you specified the option to create database files automatically, a performance library for the prior release files is created. The performance database files are moved to that library. This action allows new files to be created and preserves the existing data files from the previous release level. You should be aware of the following items:
  - The library name that is created is QPFRD*vrmnn*, where *vrm* is the current version, release, and modification and *nn* is a unique sequence number starting with 01, for example, QPFRD52001.
  -
  - The library is created with *EXCLUDE public authority. The library is owned by the QSYS user profile, and the owner of the original library is given *ALL authority.
  -
  - All QAPM*xxxx* files are moved.
  -
  - If you do not want to keep the data from the previous release, you can delete the QPFRDvrmnn library with the Delete Library (DLTLIB) command (DLTLIB qpfrdvrmnn)
-
- Delete the QAPM*xxxx* files
  If you do not convert your performance database files and do not run Collection Services, you can delete the QAPM*xxxx* files with the Delete File (DLTF) command (DLTF *library*/QAPM*).

What's new: Performance explorer

Here is what is new for this release:

- Performance explorer commands now require that users have *SERVICE authority.
- New CL commands allow you to create a filter to reduce the amount of data that you capture.
  - Add PEX Filter (ADDPEXFTR)
  - Work with PEX Filter (WRKPEXFTR)
  - Remove PEX Filter (RMVPEXFTR)
  - The STRPEX command was enhanced to support filters.
- Performance explorer now uses the management collection object (*MGTCOL) to convert collected data into its database files. The Create PEX Data (CRTPEXDTA) command performs the conversion.
- The following events were added to the Add PEX Definition (ADDPEXDFN) command:
  - Application events (APPEVT)
  - Portable Application Solution Environment (PASE) events (PASEEVT)
  - Journal events (JRNEVT)
  - iSeries NetServer, File Server, and Network File System Server and Client events (FILSVREVT)
  - Synchronization events (SYNCEVT)
  - Expert cache events (EXPCCHEVT)
- A new value was added to the Operating System events (OSEVT) parameter, *HOSTSVRCNN, on the ADDPEXDFN command. This value is the same as the *DBSVRCNN value. *HOSTSVRCNN is the recommended value.
- A new command is provided that allows you to work with your performance explorer definitions: Work with PEX Definitions (WRKPEXDFN) command.
- A new collection state of incomplete can be displayed when ending a performance explorer session. This state means that the collection ended unexpectedly.

- New database files for this release include the following:

New File	Description
QAYPEPROCI	This file contains procedure information that is associated with every instruction address in the collection data. It contains the data that was previously in the QAYPEMII, QAYPELICI, QAYPENMI, and QAYPENLIC files.
QAYPECFGI	This file contains collection definition information. It contains the data that was previously in the QAYPECOFG, QAYPECICFG, QAYPESTCFG, QAYPETRCFG, and QAYPEHWCFG files.
QAYPEFTRI	This file contains the collection filter definition information.
QAYPELTASK	This file contains the task name and number of the definition information.
QAYPERINF	This file contains miscellaneous resolution information, for example, the character string version of each IP address.
QAYPETSKSW	This file contains the information that is specific to the task switch events. It contains the data that was previously in the QAYPEBASE file.

- Deleted database files for this release include the following:

Deleted File	Where Data Moved
QAYPECICFG	QAYPECFGI
QAYPECOCFG	QAYPECFGI
QAYPEHWCFG	QAYPEPROCI
QAYPELICI	QAYPECFGI
QAYPELNAMT	QAYPELTASK
QAYPELNUMT	QAYPELTASK
QAYPEMII	QAYPEPROCI
QAYPENLIC	QAYPEPROCI
QAYPENMIC	QAYPEPROCI
QAYPEPERD	Table no longer needed because periodic mode is no longer supported.
QAYPEPSUM	The profile summary data can now be calculated from QAYPEPPANE and QAYPETASKI data.
QAYPEPWDW	This file contained profile summary data that was not used.
QAYPESTCFG	QAYPECFGI
QAYPES36	System/36 events are no longer supported.
QAYPETRCFG	QAYPECFGII
QAYPETRCPT	QAYPETIDX
QAYPEUNKWN	QAYPEUSRDF

- Changes to the database files for this release include the following:
  - Character data fields now use the default CCSID 65535.
  - Eight-byte addresses, instructions, and segments are now stored in 8-byte hexadecimal fields rather than converting each byte to 2 hexadecimal characters and stored in either 16-byte character fields or 16-byte hexadecimal fields.
  - True numeric data that is stored in packed decimal fields were changed to integer fields:
    - 5-digit packed decimal fields are changed to 9B integer fields.

- 10-digit packed decimal fields are changed to 18B integer fields.
- 20-digit packed decimal fields are changed to 18B integer fields, unless the value is greater than an 8-byte signed integer.

Print this topic

To view or download the PDF version of the performance topic, select Performance (about 350 KB or 95 pages). This PDF does not include the performance database table information.

To view or download the PDF version of the performance database table information, select Performance database tables (about 350 KB or 125 pages).

You can also view or download these related topics:

- Management Central (about 250 KB or 55 pages) includes information about how to set up your Management Central endpoint systems and system groups, as well as information about all the ways Management Central can help you streamline your server administration tasks, such as:
  - Manage users and groups
  - Package and send data
  - Run commands
  - Schedule your tasks or jobs with Management Central scheduler or Advanced Job Scheduler.
- Work Management (about 173 KB or 40 pages) describes these work management concepts:
  - A job's life
  - Daily work management
  - The structure of your system
  - How work gets done

You can also view or print any of the following PDFs:

- Manuals:
  - Performance Tools for iSeries

    (about 400 pages)
  - iSeries Performance Capabilities Reference

    Link to a web site containing the latest version of this book. This reference provides highly technical information about server performance useful for performance benchmarking, capacity planning, and planning for server performance.
  -

    This reference provides highly technical information about server performance useful for performance benchmarking, capacity planning, and planning for server performance.
- Redbooks:
  - AS/400 Performance Management

    (about 504 pages)
  - AS/400 HTTP Server Performance and Capacity Planning

(about 205 pages)

– Java and WebSphere Performance on IBM eserver iSeries Servers

(about 235 pages)

– Managing OS/400 V5R1 with Operations Navigator

(about 550 pages)

– DB2 UDB/WebSphere Performance Tuning Guide

(about 360 pages)

– Lotus Domino for AS/400: Performance, Tuning, and Capacity Planning

(about 550 pages)

– AS/400 Performance Explorer Tips and Techniques

(about 550 pages)

To save a PDF on your workstation for viewing or printing:

1. Open the PDF in your browser (click the link above).
2. In the menu of your browser, click **File**.
3. Click **Save As...**
4. Navigate to the directory in which you would like to save the PDF.
5. Click **Save**.

If you need Adobe Acrobat Reader to view or print these PDFs, you can download a copy from the Adobe Web site (www.adobe.com/prodindex/acrobat/readstep.html)

.

Plan for performance

Planning your system's performance requires you to set performance objectives, create benchmarks based on those objectives, and plan for your system's growth. This section guides you through the necessary steps in planning your system's performance.

When planning your system's performance, you will need to fully understand the business requirements your system is addressing and be able to translate those business needs into performance objectives. Keep in mind that as the business needs evolve, the performance objectives must also evolve.

Perhaps the best way to start is to estimate the maximum hourly and daily interactive transaction throughput required of your computer system during your peak business periods. After that, you can decide what average response time is acceptable to your local and remote workstations. You should think about how long your regular batch processes take, and how to schedule them so that they complete in time to achieve your business requirements.

You can then establish a base set of statistics, which should then be documented in a performance objective plan containing:

- The peak transactions per hour
- The peak transactions per day
- Acceptable average response time for local workstations
- Peak interactive transactions
- A list of the major scheduled batch jobs with times when they will be run and their expected duration
- A list of other unscheduled batch jobs that may be required

To plan for performance, complete the following tasks:

Set system benchmarks
Setting good system benchmarks will give you performance data for a properly tuned system. These performance benchmarks from both before and after system changes provide important information for both troubleshooting and planning.

Determine when and how to expand your system
As your business needs change, your system must also change. To prepare for any changes, you will want to model the current system and then see what would happen if the system, the configuration, or the workload were changed.

Capacity Upgrade on Demand
Capacity Upgrade on demand lets you add processors to your iSeries server as your changing business needs demand more resources. You can order and activate processors or a trial, temporary or permanent basis.

Select a performance management strategy
Different business needs require different performance management strategies. Here are three basic business models and their suggested performance management strategies.

Set system benchmarks

You should establish system benchmarks before any major change in the system configuration, for example adding a new interactive application or performing a system upgrade. Maintaining accurate benchmark information can provide essential troubleshooting information. At a minimum, benchmarks should include current collection objects from Collection Services. Depending on your environment you may need to maintain more detailed information using Performance Explorer.

Setting up a benchmark requires:

- That the correct iSeries configuration is available
- That the application and the data are representative and valid
- That the correct version of all programs and software to be used are available
- That the required number of users and workstations are available to run the test
- That the transactions are well defined for each user

Running meaningful benchmarks for interactive workloads is almost impossible without special equipment that allows you to simulate a user at a workstation. To run a batch benchmark is, of course, not as complex a task as to test performance of interactive applications, and the first three points above are still valid for this type of test. However, setting system benchmarks on concurrent batch and interactive work, which is frequently the actual customer environment, also requires the appropriate number of users and workstations.

Determine when and how to expand your system

As your business needs evolve, so do your system needs. To plan for future system needs and growth, you will need to determine what would happen if the system, the configuration, or the workload were changed. This process is called trend analysis and should be done monthly. As your system approaches resource capacity guidelines, you may want to gather this data more frequently.

Trend analysis should be done separately for interactive and batch environments. If your company uses a certain application extensively, you may want to perform a trend analysis for the application. Another environment that may be important to track would be the end-of-month processing. It is important that you collect trend analysis data consistently. If your system's peak workload hours are between 10:00 AM and 2:00 PM and you collect trend analysis data for this time period, do not compare this data to data collected from other time periods.

To do a proper job of capacity planning and performance analysis, you must collect, analyze, maintain, and archive performance data. IBM offers several tools that help you with your capacity planning, resource estimating, and sizing:

Performance Management/400 (PM/400)
PM/400 completely automates collecting data, analyzing data, and archiving data and provides you with summarized performance and capacity information that is easy to understand. PM/400 helps you plan for and manage system resources through ongoing analysis of key performance indicators. This function ships with the OS/400 licensed program. There is nothing that you need to do other than activate the function and periodically check that the data is being collected and transmitted to IBM. All collection sites are network secure, and the PM/400e service transmits only nonproprietary performance data to IBM. The time of the transfer is completely under your control.

Workload Estimator
The Workload Estimator is a tool that helps you size your system needs based on estimated workloads for specific workload types. Through a web-based application, you can size the upgrade to the required iSeries system that accommodates your existing system's utilization, performance, and growth as reported by PM/400. As an additional option, sizings can also include capacity for adding specific applications like Domino, Java, and WebSphere, or the consolidation of multiple AS/400 or iSeries traditional OS/400 workloads on one system. This capability allows you to plan for future system requirements based on existing utilization data coming from your own system. This application does not support the Processor On-Demand environment or an environment with logical partitions.

PATROL for iSeries (AS/400) - Predict
This product helps manage iSeries performance by automating many of the routine administration tasks required for high availability and optimal performance. Additionally, it offers detailed capacity planning information to help you plan the growth of your iSeries environment.

See Select a performance management strategy for more information about creating and implementing a performance strategy.

Capacity Upgrade on Demand

Capacity Upgrade on Demand (CUoD) for iSeries is a feature that offers you the capability to dynamically activate one or more central processors of select server models without requiring you to restart your server or interrupt your business. You can order and activate standby processors that are already installed on your server on a trial or permanent basis.

With CUoD, you can activate additional processors and pay only for the new processing power as your needs grow. You can increase your processor capacity without disrupting any of your current operations.

Use this topic to understand how CUoD works and what you need to do to take advantage of it.

Capacity Upgrade on Demand concepts
Start here for a conceptual understanding of CUoD.

Prepare for Capacity Upgrade on Demand
Consider important planning and preparation requirements.

Order Capacity Upgrade on Demand
You must order an activation feature to receive an activation code that will allow you to activate standby processors.

Activate standby processors
Use this information to activate one or more standby processors on your iSeries server.

Scenario: Capacity Upgrade on Demand
Use this information for an example of the steps an administrator might go through to plan for, order, and activate additional capacity.

Capacity Upgrade on Demand concepts

Capacity Upgrade on Demand (CUoD) provides you the capability to activate additional processors on select iSeries models on a free 14-day trial basis or by purchasing a permanent processor activation. This capability adds significant value for iSeries servers by enabling a fast and economical way to add capacity for new workloads, enabling your server to adapt to unexpected performance demands.

The select iSeries models are shipped with a number of **startup processors**. Startup processors are processors that are already active in your iSeries server when it is shipped. **Standby processors** are processors that are shipped with your server, but not available for use until you activate them. Standby processors can be temporarily activated on a 14-day trial basis or permanently activated by purchasing an activation feature and entering the provided activation code. The following table lists the number of startup and standby processors available for each model.

Model	Processor feature	Startup processors	Standby processors	Installed processors
825	2473	3	3	6
830	2349	4	4	8
830	2351*	1	7	8
840	2352	8	4	12
840	2353	12	6	18
840	2354	18	6	24
840	2416	8	4	12
840	2417	12	6	18
840	2419	18	6	24
870	2486	8	8	16
890	2487	16	8	24
890	2488	24	8	32
890	2497	16	8	24
890	2498	24	8	32

* Limited availability

The following concepts provide you with the information you need to take advantage of CUoD.

Capacity Upgrade on Demand trial period
Take advantage of the free 14-day trial period to gauge the benefits of additional processor capacity.

Software licensing considerations for Capacity Upgrade on Demand
Read about how activating CUoD affects software tiers.

Processors on demand activation code
Once you have decided to permanently activate some or all of your standby processors, you will need to order and purchase one or more Processors On Demand activation features. This will provide you with an activation code that will allow you to activate standby processors.

Electronic Service Agent requirements
Use Electronic Service Agent for quick and accurate transmission of the vital product data that is necessary to activate CUoD standby processors.

Capacity Upgrade on Demand trial period: A 14-day trial period of Capacity Upgrade on Demand is available for customers who would like to evaluate the use of standby processors at no charge. Once activated, the trial period is available for a one-time use after the initial start of the server and also for a one-time use each time an activation code is entered.

Once started, the trial period is available for a period of 14 power-on days. The trial period clock ticks only while the server is powered on. All processors, both startup and standby, are activated during the trial period. It is not possible to stop and restart the trial period.

See Activate standby processors for a trial period to try out this feature.

Software licensing considerations for Capacity Upgrade on Demand: Many business partners use the processor feature code system value available on the server to set software licensing fees for software. This system value continues to be available on servers with standby processors, but it should be noted that the processor feature code system value remains the same, no matter how many standby processors are activated.

Software providers that base their software licensing fees on the number of processors on a server have historically used a ″soft″ compliance approach. With each processor activation, it remains the customer's responsibility to inform and pay the required software licensing fees associated with any software resident on the server that has software licensed by the number of processors.

Activating a processor does not change the software tier.

Note: When a Processors on Demand activation feature is ordered through one of the IBM configurators, additional fees for IBM software licensing are billed for software products currently installed on the customer's server that are licensed by processor.

Processors On Demand activation code: When you have decided to permanently activate some or all of your standby processors, you need to order and purchase one or more Processors On Demand (POD) activation features.

When the order has been placed, the order record is combined with vital product data (VPD) that is collected from your server. This information is used to generate a POD activation code specifically for your server. You can think of this as a capacity license key for you to use to activate your standby processors.

This activation code will be posted on an IBM Web site for quick access. This usually happens within one business day (24 hours), assuming the order is accompanied with the required VPD collected from the server. Once your activation code has been generated, you can access it using your system type and serial number at the iSeries Capacity Upgrade on Demand web site:

http://www.ibm.com/servers/eserver/iseries/hardware/ondemand

For instructions on how to order POD activation features and receive POD activation codes, see Order Capacity Upgrade on Demand.

Electronic Service Agent requirements: When you place an order to purchase Capacity Upgrade on Demand, IBM combines your order information with vital product data (VPD) on your server to create the activation code necessary to unlock and activate standby processors on your server.

VPD information can be sent to IBM electronically using the Electronic Service Agent, which is part of Extreme Support, IBM's comprehensive technical service and support initiative exclusive to IBM iSeries. Electronic Service Agent is a no-charge licensed program (5798-RZG) that resides on your server and is designed to monitor events and to transmit server inventory information to IBM on a periodic, customer-definable timetable.

For complete documentation on Electronic Service Agent, including information on installation, see the Electronic Service Agent for iSeries User's Guide

Prepare for Capacity Upgrade on Demand

Capacity upgrade on Demand allows you to add processor capacity without interrupting server operation. However, to seamlessly integrate the new capacity, you should prepare your server before placing your order.

Capacity planning with Capacity Upgrade on Demand
This topic includes considerations and resources for doing capacity planning on CUoD-enabled server models.

Determine when to activate processors
This topic discusses tools that can monitor trends in resource utilization and recommends when additional capacity may be required, and how tools report CPU utilization on servers with standby processors.

Set up your environment for Capacity Upgrade on Demand
Prepare your server to integrate new processor capacity, and be ready to order additional capacity when it is required.

Capacity planning with Capacity Upgrade on Demand: Capacity planning for servers with standby processors uses essentially the same procedures and resources that are used for sizing other iSeries models. The comprehensive set of tools, resources, and offerings available to help determine the required server capacity has been updated to support servers with standby processors.

For help with capacity planning, refer to the following resources:

Determine when and how to expand your server
This topic overview discusses several tools that are available to help you with capacity planning and identifying trends in resource utilization.

iSeries Benchmark Center

Use this IBM web site for help with benchmarking application environments.

iSeries Solutions Center — Capacity Planning Services

This IBM consulting service can help you plan a server solution that meets the growing demands of your business.

Note: Workload Estimator supports CUoD, but only returns estimates for the recommended number of active processors. The additional capacity provided by the standby processors are not represented in the recommendations.

Determine when to activate processors: Capacity upgrade on Demand provides the ability to add processors to your system when your workload requires the additional resources. You should monitor your system CPU utilization and trends in CPU utilization to determine when to activate additional processors, and how many processors you will require.

There are many performance tools available to report CPU utilization information. In particular, PM/400 can identify trends in resource utilization, and iSeries Navigator monitors can give you more detailed information about how resources are being used, and alert you when utilization reaches a pre-defined critical level.

Measuring CPU utilization for servers with standby processors

When computing average usage of all available processors, the system functions reporting CPU utilization do not include the standby processors in the total amount of CPU capacity. The standby processors are not considered active within the various system functions that report CPU utilization percentages. The percentage of used CPU capacity (CPU utilization CPU percent, in iSeries Navigator) is a calculated metric based on the amount of time the processor was active within an elapsed time. This is normally reported as a percentage where 100% indicates that the processor was busy for the entire elapsed time. When multiple processors are present, CPU time must be adjusted to represent the average usage of all processors so that utilization is always reported as the percentage of total available capacity.

Interactive capability is determined by the interactive feature purchased. This capability is not impacted by the number of standby processors and does not change when standby processors become active. Interactive utilization reported as a percentage of interactive capability is not affected by the Capacity Upgrade on Demand technology. Within iSeries Navigator, this metric is called CPU Utilization (Interactive Feature).

Interactive CPU utilization is also reported as a percentage of total system CPU. Within Management Central this metric is called CPU Utilization (Interactive Jobs). This metric affects servers with CUoD in the same manner as described above for system CPU utilization.

Set up your environment for Capacity Upgrade on Demand: Before ordering any Processor On Demand (POD) activation codes, you should prepare your environment for ordering and integrating the additional capacity.

Prepare to order

When you purchase your POD activation codes, you must also provide vital product data (VPD) for your server. You can do this manually, by fax, or electronically, by using the Electronic Service Agent. Using the Electronic Service Agent to send VPD data avoids delays associated with the manual methods, and should result in the POD activation code being posted within 24 hours of receiving the order. You should allow for the time required to send and process your order when you are using additional processors on a trial basis to avoid interruptions in performance.

To use the electronic method of sending VPD data, Set up the Electronic Service Agent.

Prepare for additional capacity

To ensure your server is able to fully utilize the activated processors, you may want to make the following preparations:

- Perform any I/O conditioning
- Perform memory upgrades
- Prepare any logical partitions (LPAR)

For servers using LPAR, standby processors are always associated with the primary partition. You must assign all processors to a partition.

Set up the Electronic Service Agent: You can use the Electronic Service Agent to send VPD data. To setup the Electronic Service Agent, follow these steps:

1. Install the Electronic Service Agent.
   Refer to the Electronic Service Agent for iSeries User's Guide

   for setup and installation instructions.
2. Ensure that TCP/IP is set up and has been started.

For more information about the Electronic Service Agent, refer to Electronic Service Agent requirements.

Order Capacity Upgrade on Demand

You can order Processors on Demand (POD) activation features for a new server, a model upgrade, or an installed server. For a new server or a model upgrade, your order can contain one or more POD activation features. In this case, the POD activation code is entered before the server is shipped to you.

For an installed server, once you determine that you want to permanently activate some or all of your standby processors, you will need to order one or more POD activation features and then use the resulting POD activation code to activate your standby processors.

Notes:

1. It can take several days to process an order and post the resulting POD activation code. You can activate standby processors for a trial period of 14 days to satisfy workload requirements while your order for permanent activation of additional capacity is being fulfilled.
2. An order for POD activation features can be processed more quickly if no miscellaneous features are included with the order.

To order one or more POD activation features, follow these steps:

1. Determine the number of standby processors you want to activate.
   For information, see the Determine when to activate processors topic.
2. Contact your IBM business partner or IBM sales representative to place your order for one or more POD activation features. Customers in the United States can choose to place an order on the IBM Web site:

http://www.ibm.com

3. Send vital product data from your server to IBM .
   Your order is combined with the vital product data (VPD) collected from your server before your order is processed. The information is then used to generate a POD activation code specifically for your server. The POD activation code is mailed to you and posted on the iSeries Capacity Upgrade on Demand Web site for quick access.
4. Enter the POD activation code on your server to activate standby processors permanently.

Send vital product data to IBM: When you order one or more Processors on Demand (POD) activation features, you must provide IBM with the vital product data (VPD) for your iSeries server. You can collect and send your VPD to IBM by using the Electronic Service Agent(TM) or by printing and faxing the data.

To collect and send your VPD by using the Electronic Service Agent, follow these steps:
1. Ensure that the Electronic Service Agent is set up.
2. Start the Electronic Service Agent wizard.
3. Select **Hardware** to collect and send the required VPD information.

To print and fax your VPD, follow these steps:
1. From a system console, enter STRSST on a command line to start system service tools (SST). Sign on to SST.

   Note: To use system service tools, you need a valid service tools user ID with System capacity - administrator privileges.
2. Select option 6 (Work with System Capacity) and press Enter.
3. Select option 1 (Display System Capacity Information) and press Enter. The Display permanent system capacity display appears.
4. Press F6 to print the information.
5. Prepare a fax document with the following information:
   - Fax-To Information:
     – **Send To:** Capacity on Demand Administrator (507-253-7019)
     – **Fax number:** 507-253-4553
     – **Location:** Rochester, Minnesota
   - Fax-From Information:
     – **Customer Name:**
     – **Customer Contact Name:**
     – **Customer Address:**
     – **Customer Phone Number:**
     – **Customer Fax Number:**
6. Fax the VPD to the following fax number:
   507-253-4553

Activate standby processors
You can activate one or more standby processors that are already installed on your server on a trial or permanent basis.

Activate standby processors for a trial period
You can activate all of the standby processors installed on your server for a trial period of 14 days. This allows you to evaluate how the additional capacity can benefit your environment before deciding

whether to purchase additional capacity or to satisfy peak workload requirements while your order for the permanent activation of additional capacity is being fulfilled.

Activate standby processors permanently
You can permanently activate a specified number of standby processors.

Activate standby processors for a trial period: To activate all of the standby processors installed on your server for a trial period, follow these steps:

1. From a system console, enter STRSST on a command line to start system service tools (SST). Sign on to SST.

   Note: To use system service tools, you need a valid service tools user ID with System capacity - administrator privileges.
2. Select option 6 (Work with system capacity) and press Enter. The Work with System Capacity display appears.
3. Select option 3 (Work with temporary system capacity) and press Enter. The Confirm start temporary system capacity activation display appears.
   Note: This option will not appear if temporary system capacity activation is in use or has already been used.
4. Press Enter to confirm the system capacity activation.
5. Exit the Work with System Capacity display.
6. If the server is partitioned, you must assign the newly available processors to a logical partition before you can use the new capacity.

   For information on assigning the activated processors to a logical partition, see Dynamic movement of processing power.
7. If the server is not partitioned, follow these steps:
   a. Select option 5 (Work with system partitions) and press Enter.
   b. Select option 3 (Work with partition configuration) and press Enter.
   c. Type 2 (Change partition processing resources) next to PRIMARY and press Enter.
   d. Enter a value for the New number of processors that represents the total number of active processors.

      Note: A configuration error at the bottom of the display indicates that the default Primary partition has been altered (logical partitioning is actively being used), and you must refer to your logical partition plan to properly assign the activated processors to the server.
   e. Press Enter to confirm the change.
8. Exit SST.

You can now begin using the new capacity.

Notes:
1. The trial automatically ends after 14 days have elapsed or when a new Processors on Demand (POD) activation code is entered.
2. If your server is partitioned, you must return the standby processors to the primary partition at the end of the trial period.

Activate standby processors permanently: When you have purchased one or more Processors on Demand (POD) activation features, you will receive a POD activation code to activate your standby processors.

To permanently activate some or all of your standby processors, follow these steps:

1. Retrieve the POD activation code as follows:
   a. Access the iSeries Capacity Upgrade on Demand Web site: http://www.ibm.com/servers/eserver/iseries/hardware/ondemand
   b. Enter the system type and serial number of your server.
   c. Record the POD activation code displayed on the Web site.
2. Enter STRSST on a command line to start system service tools (SST). Sign on to SST.

   Note: To use system service tools, you need a valid service tools user ID with System capacity - administrator privileges.
3. Select option 6 (Work with system capacity) and press Enter.
4. Select option 2 (Activate permanent system capacity) and press Enter.
5. Enter your POD activation code in the Processor on Demand activation code field and press Enter. The Confirm Activate System Capacity display appears.
6. Press Enter to confirm the system capacity activation.
7. Exit the Work with System Capacity display.
8. If the server is partitioned, you must assign the newly available processors to a logical partition before you can use the new capacity.

   For information on assigning the activated processors to a logical partition, see Dynamic movement of processing power.
9. If the server is not partitioned, follow these steps:
   a. Select option 5 (Work with system partitions) and press Enter.
   b. Select option 3 (Work with partition configuration) and press Enter.
   c. Type 2 (Change partition processing resources) next to PRIMARY and press Enter.
   d. Enter a value for the New number of processors that represents the total number of active processors.
      Note: A configuration error at the bottom of the display indicates that the default Primary partition has been altered (logical partitioning is actively being used), and you must refer to your logical partition plan to properly assign the activated processors to the server.
   e. Press Enter to confirm the change.
10. Exit SST.

Note: When you enter a POD activation code, the standby processors immediately become active. However, the server must run for 15 minutes to store the POD activation code. If you enter a POD activation code and shut down the server before it has run for 15 minutes, you may have to re-enter the POD activation code when you start the server.

You can now begin using the new capacity.

Scenario: Capacity Upgrade on Demand

Capacity Upgrade on Demand allows customers to activate standby processors as their workload requires it. The following scenario walks through the steps of planning for, ordering, and leveraging this feature.

1. A Model 840 server with feature-code 2416 is operating with eight active processors and four standby processors. As the server workload grows, the available CPU resource utilization consistently approaches or exceeds 70% of the available capacity. Anticipating the need for additional resources, the administrator decides to consider activating some of the standby processors.

2. Before activating any processors, the administrator prepares the server for Capacity Upgrade on Demand. This involves performing trend analysis to learn how many additional processors will be required, preparing the server to leverage additional processors, and preparing to order the new capacity.
3. To investigate the benefits of activating the additional processors, the administrator decides to activate the processors for a trial period. The trial period lasts 14 days.
4. After deciding that the performance improvement gained by activating the additional processors warrants purchasing the processors permanently, the administrator contacts the IBM sales representative or business partner, or visits www.ibm.com to place an order for four Processor on Demand (POD) activation codes.
5. The IBM sales representative places the order in the IBM configurator, and receives a reminder to send the vital product data (VPD) from the server with the order. The VPD data may be faxed to IBM or sent electronically with the Electronic Service Agent.
6. The administrator retrieves the POD activation codes from the web, and activates the permanent capacity. This involves entering the POD activation code on the target server and assigning the processors to a logical partition.

The Model 840 now has all eight processors available for use.

Select a performance management strategy

Developing a good performance management strategy will help you manage your system's performance. Your performance management strategy depends in a large part on the amount of time you can afford to spend managing performance. If you are working with a small company, you may be managing many different aspects of your business and cannot devote many hours to managing performance. Many large companies employ performance specialists to keep their systems tuned and running effectively.

For determining a basic performance management strategy and for identifying which performance applications to use, classify your company in one of three categories: small business, mid-sized business, and large business. The business resources vary for each size, and your management strategy will vary accordingly.

Small business
A small business most likely has fewer resources to devote to managing performance than a larger business. For that reason, use as much automation as possible. You use PM/400 to have your performance data sent directly to IBM where it will be compiled and generated into reports for you. This not only saves you time, but IBM also makes suggestions to you when your iSeries server needs an upgrade.

The following is a list of recommended performance applications for a small business:

Collection Services
Collect sample data at user-defined intervals for later analysis.

Graph History
Display performance data collected with Collection Services.

PM/400
Automate the collection, archival, and analysis of system performance data.

Performance Tools
Gather, analyze, and maintain system performance information.

Monitors

Observe graphical representations of iSeries system perforance, and automate responses to predefined events or conditions.

Mid-sized business

The mid-sized business probably has more resources devoted to managing performance than the small business. You may still want to automate as much as possible and can also benefit from using PM/400.

The following is a list of recommended performance applications for a mid-sized business:

Collection Services
Collect sample data at user-defined intervals for later analysis.

Graph History
Display performance data collected with Collection Services.

PM/400
Automate the collection, archival, and analysis of system performance data.

Performance Tools
Gather, analyze, and maintain system performance information.

Monitors
Observe graphical representations of iSeries system perforance, and automate responses to predefined events or conditions.

Performance explorer
Collect detailed information about a specific application or system resource.

Large business

The large business has resources devoted to managing performance.

The following is a list of recommended performance applications for a large business:

Collection Services
Collect sample data at user-defined intervals for later analysis.

Graph History
Display performance data collected with Collection Services.

PM/400
Automate the collection, archival, and analysis of system performance data.

Performance Tools
Gather, analyze, and maintain system performance information.

Monitors
Observe graphical representations of iSeries system perforance, and automate responses to predefined events or conditions.

Performance explorer
Collect detailed information about a specific application or system resource.

iDoctor for iSeries

Analyze trace data to improve system and application performance.

Performance Trace Data Visualizer (PTDV)
View trace data from a Java application.

Set up your environment to manage performance

The iSeries server includes several tools that regularly collect system performance data and monitor your system for performance trends and potential problems. Your unique requirements and environment will determine both the tools you choose to invest in and the configuration choices you should make. Effectively setting up your system will allow you to do accurate capacity planning as your system grows and to resolve performance problems when they occur.

Use the following topics to learn about and configure tools that will collect, monitor, and analyze your system performance.

Collection Services
Collection Services manages the routine collection of your system performance data. This tool regularly collects data and creates archives called collection objects. These collection objects may be accessed directly by some tools or converted into sets of database files for analysis with your own custom queries or by other tools and reports. Because Collection Services mainly provides data for other applications, the other tools you are using will significantly affect your configuration choices, including how frequently you collect data, the types of data you collect, and the length of time you will keep the data on your system.

Performance Management/400
PM/400 uses Collection Services to gather non-proprietary performance data, and sends it to IBM for storage and expert analysis. This eliminates the need to store and maintain it yourself. You can then access detailed reports and recommendations about your system's performance with a web browser.

iSeries Navigator monitors
The monitors included in iSeries Navigator use Collection Services data to track the elements of system performance of specific interest to you. Moreover, they can take specified actions when certain events, such as the percentage of CPU utilization or the status of a job, occur. Use this topic to learn how to use these monitors and how to set them up on your system.

Manage iSeries performance

Successfully managing performance ensures that your system is efficiently using resources and that your iSeries server provides the best possible services to your users and to your business needs. Moreover, effective performance management can help you quickly respond to changes in your system and can save you money by postponing costly upgrades and service fees.

Understanding the factors that affect system performance helps you respond to problems and make better long-term plans. Effective planning can prevent potential performance problems from developing and ensures that you have the system capacity to handle your current and growing workloads.

Use the following topics to learn how to maintain your system's performance and to resolve performance problems.

Track performance
Tracking your system performance over time allows you to plan for your system's growth and ensures that you have data to help isolate and identify the cause of performance problems. Learn which applications to use and how to routinely collect performance data.

Research a performance problem
There are many options available to help you identify and resolve performance problems. Learn how to use the available tools and reports that can help you find the source of the performance problem.

Display performance data
After you have collected performance data, learn how to display the data using the most appropriate tool for your purposes.

Tune performance
When you have identified a performance problem, you will want to tune the system to fix it.

Manage e-business performance
Managing performance in an e-business environment introduces several new problems for the iSeries administrator. Refer to this topic for information and resources to help you plan for, track, and improve performance for your e-business applications.

Track performance

Tracking system performance for the iSeries server helps you identify trends that can help you tune your system configuration and make the best choices about when and how to upgrade your system. Moreover, when problems occur, it is essential to have performance data from before and after the incident to narrow down the cause of the performance problem, and find an appropriate resolution.

The iSeries server includes several applications for tracking performance trends and maintaining a historical record of iSeries performance data. Most of these applications use the data collected by Collection Services. You can use Collection Services to watch for trends in the following areas:

- Trends in system resource utilization. You can use this information to plan and specifically tailor system configuration changes and upgrades.
- Identification of stress on physical components of the configuration.
- Balance between the use of system resource by interactive jobs and batch jobs during peak and normal usage.
- Configuration changes. You can use Collection Services data to accurately predict the effect of changes like adding user groups, increased interactive jobs, and other changes.
- Identification of jobs which may be causing problems with other activity on the system

The following tools will help you monitor your system performance over time:

Collection services
Collection services gathers performance data at user-defined time intervals and then stores this information in collection objects on your system. Many of the other tools, including monitors, Graph history, PM/400, and many functions in the Performance Tools licensed program, rely on these collection objects for their data.

Graph history
Graph history displays performance data collected with Collection Services over a specified period of time through a graphical user interface (GUI). The length of time available for display depends on how long you are retaining the collection objects, and whether you are using PM/400.

PM/400
PM/400 automates the collection, archival, and analysis of system performance data and returns clear reports to help you manage system resources and capacity.

Research a performance problem

Most of the tools that collect or analyze performance use either trace or sample data. Collection Services regularly collects sample data on a variety of system resources. Several tools analyze or report on this sample data, and you can use this to get a broader view of system resource utilization and to answer many common performance questions. For more detailed performance information, several tools generate trace-level data. Often, trace-level data can provide detailed information about the behavior and resource consumption of jobs and applications on your system. Performance Explorer and the STRPFRTRC command are two common tools for generating trace data.

For example, if your system is running slowly, you might use the system monitor to look for problems. If you see that the CPU utilization is high, you could identify any jobs that seem to be using an unusually large amount of resources. Then, you may be able to correct the problem by making configuration changes. However, some problems will require additional information. To get detailed information about that job's performance you could start a performance explorer session, gather detailed information about that job's behavior on the iSeries system, and potentially make changes to the originating program.

To learn more about collecting performance data, use the following topics to learn how and when to use some of the performance management applications.

Identify a performance problem
Learn the common steps involved with identifying a performance problem.

Identify and resolve common performance problems
Many different performance problems often affect common areas of the iSeries system. Learn how to research and resolve problems in common areas.

Collect system performance information
Collection Services regularly collects information about system performance. Often, analyzing performance data begins with this information.

Collect information about system resource utilization
Several tools monitor how resources like CPU, disk space, interactive capacity, and many other elements, are being used. You can use these tools to start identifying problem areas.

Collect information about an application's performance
An application may be performing slowly for a variety of reasons. You can use several of the tools included in OS/400 and other licensed programs to help you get more information.

Scenario: Improve system performance after an upgrade or migration
In this scenario, you have just upgraded or migrated your system and it now appears to be running slower than before. This scenario will help you identify and fix your performance problem.

Identify a performance problem

When trying to identify a performance problem, it is important to assess whether the hardware configuration is adequate to support the workload. Is there enough CPU capacity? Is the main storage sufficient for the different types of applications? Answering these questions first, perhaps through capacity modeling techniques, prevents needless effort later.

With an understanding of the symptoms of the problem and the objectives to be met, the analyst can formulate a hypothesis that may explain the cause of the problem. The analyst can use commands and tools available with the OS/400 operating system and the Performance Tools licensed program to measure the system performance.

Reviewing the measured data helps to further define the problem and helps to validate or reject the hypothesis. Once the apparent cause or causes have been isolated, a solution can be proposed. When you handle one solution at a time, you can redesign and test programs. Again, the analyst's tools can, in many cases, measure the effectiveness of the solution and look for possible side effects.

To achieve optimum performance, you must recognize the interrelationship among the critical system resources and attempt to balance these resources, namely CPU, disk, main storage, and for communications, remote lines. Each of these resources can cause a performance degradation.

Improvements to system performance, whether to interactive throughput, interactive response time, batch throughput, or some combination of these, may take many forms, from simply adjusting activity level or pool size to changing the application code itself. In this instance, an activity level is a characteristic of a subsystem that specifies the maximum number of jobs that can compete at the same time for the processing unit.

Identify and resolve common performance problems

When performance problems occur on the iSeries server, they often affect certain areas of the system first. Refer to the following table for some methods available for researching performance on these system areas. Many of these areas are available as system monitor metrics. However, there are several other ways to access information about them.

Area	Description	Available tools
Processor load	Determine if there are too many jobs on the system or if some jobs are using a large percentage of processor time.	• Work with Active Jobs (WRKACTJOB) command. • Work with System Activity (WRKSYSACT) command, which is part of Performance Tools licensed program. • The work management function in iSeries Navigator. • CPU utilization metrics within the iSeries Navigator system monitor.
Main storage	Investigate faulting and the wait-to-ineligible transitions.	• Work with Disk Status (WRKDSKSTS) command • Disk storage metrics within the iSeries Navigator system monitor • Work with System Status (WRKSYSSTS) command • The Memory Pools function under Work Management in iSeries Navigator.
Disk	Determine if there are too few arms or if the arms are too slow.	• Work with Disk Status (WRKDSKSTS) command. • Disk arm utilization metrics within the iSeries Navigator system monitor. • Performance Tools System and Component report.
Communications	Find slow lines, errors on the line, or too many users for the line.	• Performance Tools Component Report. • LAN utilization metrics within the iSeries Navigator system monitor.
IOPs	Determine if any IOPs are not balanced or if there are not enough IOPs.	• Performance Tools Component Report. • IOP utilization metrics within the iSeries Navigator system monitor.

Area	Description	Available tools
Software	Investigate locks and mutual exclusions (mutexes).	• Performance Tools Locks report • Performance Tools Trace report • Work with Object Locks (WRKOBJLCK) command. • In iSeries Navigator, right-click on the suspect job under Work Management, and select **Details —> Locked Objects**.

Collect system performance data

Collecting data is an important step toward improving performance. When you collect performance data, you gather information about your server that can be used to understand response times and throughput. It is a way to capture the performance status of the server, or set of servers, involved in getting your work done. The collection of data provides a context, or a starting point, for any comparisons and analysis that can be done later. When you use your first data collections, you have a benchmark for future improvements and a start on improving your performance today. You can use the performance data you collect to make adjustments, improve response times, and help your systems achieve peak performance. Performance problem analysis often begins with the simple question: “What changed?” Performance data helps you answer that question.

You can use Collection Services to collect performance data, create performance files with the Create Performance Data (CRTPFRDTA) command, convert them to current release with Convert Performance Data (CVTPFRDTA) command or through the Performance Tools plugin in iSeries, and then create reports or create your own queries by using the information in the performance database files.

For more information about performance data, see the following:

Collection Services
See how to collect performance data for analysis and how to customize your collections.

Performance database files
Find an overview of the performance database files that are available to you and see detailed field data for each performance database file.

In addition, you can use the Performance Management APIs to start, end, and cycle collections, as well as to change and retrieve system parameters for the data collected. For more information about other iSeries performance tools and techniques, see the Performance page.

Collect information about system resource utilization

Many tools are available to help you monitor and track the way the iSeries server and your applications are using the available resources. You can use this information as a starting point for problem analysis, and to identify trends that will help you with capacity planning and managing the growth of your system.

See the following topics to learn how and when to use these tools:

iSeries Navigator monitors
The monitors included in iSeries Navigator provide current and recent data on a wide variety of metrics. Additionally, you can configure them to take a specified action when certain events occur.

OS/400 performance commands
OS/400 includes several important functions to help you manage and maintain system performance.

PM/400
PM/400 uses Collection Services to gather non-proprietary performance data and sends it to IBM for

storage and expert analysis. This eliminates the need to store and maintain it yourself. You can then access detailed reports and recommendations about your system's performance and trend analysis with a web browser.

Collect information about an application's performance

Collecting information about an application's performance is quite different from collecting information about system performance. Collecting application information can only be done with certain performance applications such as Performance Explorer, Performance Trace Data Visualizer (PTDV), and iDoctor. Alternately, you can get an overview of application performance by using the Job monitor to track individual server performance and Performance Tools to track and analyze server jobs.

Note: Collecting an application's performance data can significantly affect the performance of your system. Before beginning the collection, make sure you have tried all other collection options.

Performance Explorer
This tool helps find the causes of performance problems that cannot be identified by using tools that do general performance monitoring. As your computer environment grows both in size and in complexity, it is reasonable for your performance analysis to gain in complexity as well. The performance explorer addresses this growth in complexity by gathering data on complex performance problems.

Performance explorer is designed for application developers who are interested in understanding or improving the performance of their programs. It is also useful for users who are knowledgeable in performance management to help identify and isolate complex performance problems.

Performance Trace Data Visualizer for iSeries (PTDV)
This tool is a Java application that can be used for performance analysis of applications running on iSeries. PTDV works with the performance explorer function of the OS/400 operating system to allow the analyst to view program flows and get details (such as CPU time, current system time, number of cycles, and number of instructions) summarized by trace, job, thread, and procedures. When visualizing Java application traces, additional details such as the number and type of objects created, as well as information about Java locking behavior, can be displayed. There is also support for performance explorer events generated by the WebSphere Application Server. PTDV allows sorting of columns, exporting of data, and many levels of data summarization.

For more information, go to the Performance Trace Data Visualizer

Web site.

iDoctor for iSeries
The Performance Explorer Analyzer function in iDoctor includes a software tool specifically geared towards analyzing trace data to improve system and application performance. This detailed analysis gives a low-level summary of disk operations, CPU utilization, file-open operations, machine interface (MI) programs, wait states, disk space consumption, and much more. The client component is an iSeries Navigator plugin that allows a user to condense and display iSeries trace data graphically.

Start Performance Trace (STRPFRTRC) command
OS/400 includes a command to collect multi-programming and transaction data. This command collects the data that STRPFRMON collected in previous releases. After running this command, you can export the data to a database file with the Dump trace (DMPTRC) command.

Dump trace data: Deciding when to dump trace data is a significant decision because the dump affects system performance. The Dump Trace (DMPTRC) command puts information from an internal trace table into a database file. It is not good to dump trace data during peak activity on a loaded system or within a high priority (interactive) job. You can delay a trace dump, but you want to dump the data before you forget that it exists. If the trace table becomes cleared for any reason, you lose the trace data. However, delaying the dump slightly and then using the DMPTRC command to dump the trace in a batch job can preserve performance for the users.

To dump trace data, issue the following command:

```
DMPTRC MBR(member-name) LIB(library-name)
```

You must specify a member name and a library name in which to store the data. You can collect sample-based data with Collection Services at the same time that you collect trace information. When you collect sample data and trace data together like this, you should place their data into consistently named members. In other words, the names that you provide in the CRTPFRDTA TOMBR and TOLIB parameters should be the same as the names that you provide in the DMPTRC MBR and LIB parameters.

Scenario: Improve system performance after an upgrade or migration

You recently upgraded your iSeries server to the newest release. After completing the upgrade and resuming normal operations, your system performance has decreased significantly. You would like to identify the cause of the performance problem and restore your system to normal performance levels.

Isolate changes

Several problems may result in decreased performance after upgrading the operating system. You can use the performance management tools included in OS/400 and Performance Tools licensed program (5722-PT1) to get more information about the performance problem and to narrow down suspected problems to a likely cause.

1. Check CPU utilization. Occasionally, a job will be unable to access some of its required resources after an upgrade. This may result in a single job consuming an unacceptable amount of the CPU resources.
   - Use WRKSYSACT, WRKSYSSTS, WRKACTJOB, or iSeries Navigator system monitors to find the total CPU utilization.
   - If CPU utilization is high, for example, greater than 90%, check the amount of CPU utilized by active jobs. If a single job is consuming more than 30% of the CPU resources, it may be missing file calls or objects. You can then refer to the vendor, for vendor-supplied programs, or the job's owner or programmer for additional support.
2. Start a performance trace with the STRPFRTRC command, and then use the system and component reports to identify and correct the following possible problems:
   - If the page fault rate for the machine pool is higher than 10 faults/second, give the machine pool more memory until the fault rate falls below this level.
   - If the disk utilization is greater than 40%, look at the waiting and service time. If these values are acceptable, you may need to reduce workload to manage priorities.
   - If the IOP utilization is greater than 60%, add an additional IOP and assign some disk resource to it.
   - If the page faults in the user pool are unacceptably high, refer to the topic Automatically tune performance.
3. Run the job summary report and refer to the **Seize lock conflict report**. If the number of seize or lock conflicts is high, ensure that the access path size is set to 1TB. If the seize or lock conflicts are on a user profile, and if the referenced user profile owns many objects, reduce the number of objects owned by that profile.
4. Run iDoctor with the **Task switch** option for five minutes. Then analyze the resulting trace data with the task switch monitor. Identify and resolve any of the following:
   - Jobs waiting for CPU
   - Jobs faulting
   - Seize conflicts

Display performance data

Displaying performance data helps you analyze your system's performance more accurately. Performance data can be displayed in many different ways; however, you may find a certain performance application more appropriate in some situations. Most applications display data collected with either Collection Services or from a performance trace. The best way to access that data depends on whether you are attempting to resolve a performance problem, are monitoring your system performance to plan for future growth, or are identifying trends.

Display near real-time performance data

Use the following tools to display current or very recent performance information:

OS/400 commands
There are many commands in the base operating system that will let you view current information about specific areas of system performance.

Performance Tools display
The Performance Tools licensed program includes a plugin for iSeries Navigator that displays performance data from Collection Services collection objects. You can also view detailed information about the jobs on the system and print Performance Tools reports.

System and Job monitor
These monitors display performance data for many system elements. Monitor data is based on the collection objects, and will display data as it is collected, according to the collection interval in Collection Services.

Display historical performance data

Use the following tools to view data that is stored on your system:

PM/400
PM/400 automates the collection, archival, and analysis of system performance data and returns clear reports to help you manage system resources and capacity.

Graph history
Graph history provides a graphical display of up to a week's worth of performance data depending on the retention period in Collection Services. With PM/400, Graph History can display much longer periods of data collection.

Tune performance

The primary aim of performance tuning is to allow your servers to make the best use of the system resources, and to allow workloads to run as efficiently as possible. Performance tuning is a way to adjust the performance of the system either manually or automatically. Many options exist for tuning your system. Each system environment is unique in that it requires you to observe performance and make adjustments that are best for your environment; in other words, you are required to do routine performance monitoring. For more information about the performance monitoring steps that must precede performance tuning, see Manage performance.

IBM also offers a tool that allows you to improve both the I/O subsystem and system response times by reducing the number of physical I/O requests that are read from disk. Learn how you can improve your system performance with Extended Adaptive Cache.

For more information about performance tuning, select from the following topics:

Basic performance tuning
To tune your system’s performance, you need to set up your initial tuning values, observe the system performance, review the values, and determine what to tune.

Automatically tune performance
Most users should set up the system to make performance adjustment automatically. When new systems are shipped, they are configured to adjust automatically.

Basic performance tuning

To begin tuning performance, you must first set initial tuning values by determining your initial machine and user pool sizes. Then, you can begin to observe the system performance.

Set initial tuning values

Setting initial tuning values includes the steps you take to initially configure the system pool sizes and activity levels to tune your system efficiently. The initial values are based on estimates; therefore, the estimates may require further tuning while the system is active. The following steps set the initial tuning values:

- Determine initial machine pool size
- Determine initial user pool sizes

Observe system performance

To observe the system performance, you can use the Work with System Status (WRKSYSSTS), Work with Disk Status (WRKDSKSTS), and Work with Active Jobs (WRKACTJOB) commands. With each observation period, you should examine and evaluate the measurements of system performance against your performance goals.

1. Remove any irregular system activity. Irregular activities that may cause severe performance degradation are, for example, interactive program compilations, communications error recovery procedures (ERP), open query file (OPNQRYF), application errors, and sign-off activity.
2. Use the WRKSYSSTS, WRKDSKSTS, and WRKACTJOB commands to display performance data. You can also use the Performance Tools command, Work with System Activity (WRKSYSACT), to display performance data.
3. Allow the system to collect data for a minimum of 5 minutes.
4. Evaluate the measures of performance against your performance goals. Typical measurements include:
   - Interactive throughput and response time, available from the WRKACTJOB display.
   - Batch throughput. Observe the auxiliary input/output (AuxIO) and CPU percentage (CPU%) values for active batch jobs.
   - Spooled throughput. Observe the auxiliary input/output (AuxIO) and CPU percentage (CPU%) values for active writers.
5. If you observe performance data that does not meet your expectations, tune your system based on the new data. Be sure to:
   - Measure and compare all key performance measurements.
   - Make and evaluate adjustments one at a time.

Review performance

Once you have set good tuning values, you should periodically review them to ensure your system continues to do well. Ongoing tuning consists of observing aspects of system performance and adjusting to recommended guidelines.

To gather meaningful statistics, you should observe system performance during typical levels of activity. For example, statistics gathered while no jobs are running on the system are of little value in assessing system performance. If performance is not satisfactory in spite of your best efforts, you should evaluate the capabilities of your configuration. To meet your objectives, consider the following:

- Processor upgrades
- Additional storage devices and controllers
- Additional main storage
- Application modification

By applying one or more of these approaches, you should achieve your objectives. If, after a reasonable effort, you are still unable to meet your objectives, you should determine whether your objectives are realistic for the type of work you are doing.

Determine what to tune
If your system performance has degraded and needs tuning, refer to Research a performance problem to identify the source of the performance problem and to make specific corrections.

Automatically tune performance

The system can set performance values automatically to provide efficient use of system resources. You can set up the system to tune system performance automatically by:

- Adjusting storage pool sizes and activity levels
- Adjusting storage pool paging

Adjusting storage pool sizes and activity levels
Use the QPFRADJ system value to control automatic tuning of storage pools and activity levels. This value indicates whether the system should adjust values at system restart (IPL) or periodically after restart.

You can set up the system to adjust performance at IPL, dynamically, or both.

- To set up the system to tune only at system restart (IPL), select Configuration and Service -> System Values -> Performance in iSeries Navigator. Click the Memory Pools tab and select **At system restart** under **Automatically adjust memory pools and activity levels**. This is equivalent to setting the QPFRADJ system value to 1.
- To set up the system to make storage pool adjustments at system restart (IPL) and to make storage pool adjustments periodically after restart, select Configuration and Service -> System Values -> Performance in iSeries Navigator. Click the Memory Pools tab and select both **At system restart** and **Periodically after restart** under **Automatically adjust memory pools and activity levels**. This is equivalent to setting the QPFRADJ system value to 2.
- To set up the system to make storage pool adjustments periodically after restart and not at system restart (IPL), select Configuration and Service -> System Values -> Performance in iSeries Navigator. Click the Memory Pools tab and select **Periodically after restart** under **Automatically adjust memory pools and activity levels**. This is equivalent to setting the QPFRADJ system value to 3.

The storage pool values are not reset at system restart (IPL) to the initial values.

Adjusting storage pool paging
The dynamic tuning support provided by the system automatically adjusts pool sizes and activity levels for shared pools to improve the performance of the system. This tuning works by moving storage from storage pools that have minimal use to pools that would benefit from more storage. This tuning also sets activity levels to balance the number of threads in the pool with the storage allocated for the pool. To adjust the system, the tuner uses a guideline that is calculated based on the number of threads.

When dynamic adjustment is in effect, the following performance values are changed automatically to the appropriate settings:

- Machine (*MACHINE) memory pool size (QMCHPOOL system value)
- Base (*BASE) memory pool activity level (QBASACTLVL system value)
- Pool size and activity level for the shared pool *INTERACT
- Pool size and activity level for the shared pool *SHRPOOL

- Pool sizes and activity levels for the shared pools *SHRPOOL1-*SHRPOOL60

When dynamic adjustment is in effect (the QPFRADJ system value is set to 2 or 3), the job QPFRADJ that runs under profile QSYS is seen as active on the system.

For more information about memory pools, see Memory Pools.

Manage e-business performance

Performance in an e-business environment presents several complex problems to the iSeries system administrator. In addition to routine tuning on the iSeries server, administrators must also monitor and optimize the hardware and services supporting their e-business transactions.

The following topics can help you become familiar with some of the important considerations for maximizing your server's e-business performance, and will provide links to additional resources for detailed recommendations and examples.

Client performance
While the iSeries system administrator often has little control of the client-side of the e-business network, you can use these recommendations to ensure that client devices are optimized for an eBusinsess environment.

Network performance
The network design, hardware resources and traffic pressure often have a significant effect on the performance of e-business applications. You can use this topic for information on how to optimize network performance, and tune iSeries communication resources.

Java performance in OS/400
OS/400 provides several configuration options and resources for optimizing the performance of Java applications or services on the iSeries server. Use this topic to learn about the Java environment in OS/400, and how to get the best possible performance from Java-based applications.

HTTP Server performance
The HTTP server is often an important part of e-business performance on the iSeries server. IBM provides several options and configuration choices that allow you to get the most out of this server.

WebSphere performance
WebSphere Application Server is the e-business application deployment environment of choice for the iSeries server. Use this topic to learn how to plan for and optimize performance in a WebSphere environment.

In addition to these specific recommendations, administrators should also be familiar with the following topics:

- Work management
- Java for iSeries
- HTTP server
- Domino for iSeries sizing and performance tuning

Client performance

Clients consisting of a PC with a web browser often represent the eBussiness component that administrators have the least direct control over. However, these components still have a significant affect on the end-to-end response time for web applications.

To help ensure high-end performance, client PCs should:

- Have adequate memory. Resource intensive applets, and interfaces using complex forms and graphics may also place demands on the client's processor.
- Utilize a high-speed and optimized network connection. Many communication adapters on a client PC may function while they are not optimized for their network environment. For more information refer to the documentation for your communication hardware.
- Use browsers that fully support the required technologies. Moreover, browser support and performance should be a major concern when designing the web interface.

Network performance

The network often plays a major role in the response time for web applications. Moreover, the performance impact for network components is often complex and difficult to measure because network traffic and the available bandwidth may change frequently and are affected by influences the system administrator may not have direct control over. However, there are several resources available to help you monitor and tune the communication resources on your iSeries server.

Refer to the following topics for more information:

Collection Services
Collection Services collects performance data for communication resources at regular intervals. Of particular interest, the performance data files QAPMTCP and QAPMTCPIFC store information about TCP servers. You can reference this data by querying the files directly, or by using the reports included in the Performance Tools licensed program.

System monitor
You can use the system monitors to provide information about how system resources, including communications hardware, are being used on an iSeries server. In particular, the line utilization and IOP metrics within the system monitor can provide valuable data about network performance.

Track performance
Several applications and tools allow you to routinely collect performance data for communication resources on the iSeries server and to monitor their performance over time.

iSeries Performance Capabilities Reference

The Performance Capabilities Reference provides detailed information, reports and examples that can help you configure or tune your iSeries server for optimal performance. In particular, see chapter 5: Communications Performance to help you plan for and manage iSeries communication resources.

iSeries Network.com

This web site hosts many resources for optimizing your network plan and resources. In particular, refer to the articles “Cultivate your AS/400 Networks” and “8 tools for better network performance.”

Java performance in OS/400

Java is often the language of choice for web-based applications. However, Java applications may require some optimization, both of the OS/400 execution environment and of the Java application, to get optimal performance.

Use the following resources to learn about the Java environment in OS/400, and the available tips and tools for analyzing and improving Java performance.

Java performance
There are several important configuration choices and tools to help you get the best performance from Java-based applications.

Collect information about an application's performance
There are several tools available to help you monitor and tune an applications performance in OS/400. Use this topic to learn how to use performance traces, performance explorer (PEX), and similar tools, to help you measure and improve application performance.

iSeries Performance Capabilities Reference

The Performance Capabilities Reference provides detailed information, reports and examples that can help you configure or tune your iSeries server for optimal performance. In particular, see chapter 7: Java Performance, to help you optimize the performance of Java applications, and learn performance tips for programming in Java.

Java and WebSphere performance in OS/400

Use this Redbook to learn how to plan for and configure your operating environment to maximize Java and WebSphere performance, and to help you collect and analyze performance data.

WebSphere J2EE application development for the IBM eServer iSeries server

This Redbook provides in introduction to J2EE, and offers suggestions and examples to help you successfully implement J2EE applications on the iSeries server.

In addition to performance information, the Java topic provides resources for developing and deploying Java applications on the iSeries server.

HTTP Server performance

HTTP server can play an important role in the end-to-end performance of your web-based applications, and several new improvements allow you to effectively monitor and improve web server performance. In particular the new Fast Response Caching Accelerator (FRCA) may allow you to significantly improve HTTP server performance, particularly in predominantly static environments.

Refer to the following resources for information on how to maximize HTTP server performance.

Collection Services
You can use Collection Services to collect HTTP server performance data and monitor the results over time. The performance data files QAPMHTTPB and QAPMHTTPD store HTTP server data for each collection interval. QAPMHHTB provides basic information, while QAPMHTTPD provides more detailed statistics. You can query these data files directly, or refer to the System and Component reports in the Performance Tools Licensed Program.

IBM HTTP Server for iSeries
Refer to this topic for information on setting up, configuring, and managing an HTTP server on the iSeries. This topic also includes descriptions of the latest enhancements, like the Fast Response Caching Accelerator (FRCA), to this product.

iSeries Performance Capabilities Reference

The Performance Capabilities Reference provides detailed information, reports, and examples that can help you configure or tune your iSeries server for optimal performance. In particular, see chapter 6: Web Server and Web Commerce, for HTTP server performance specifications, planning information, and performance tips.

HTTP server (Powered by Apache)

Use this Redbook to get an in-depth description of HTTP Server (Powered by Apache) on OS/400, including examples for configuring HTTP Server in common usage scenarios.

AS/400 HTTP Server Performance and Capacity Planning

Use this Redbook to learn about HTTP server impacts on performance tuning and planning. This publication also includes suggestions for using iSeries performance management tools to collect, interpret, and respond to web server performance data.

WebSphere performance

Managing iSeries server performance in a WebSphere environment presents several challenges to the iSeries administrator. Web-based transactions may consume more resources, and consume them differently than traditional communication workloads.

Refer to the following topics and resources to learn how to plan for optimal performance, and adjust server resources in a WebSphere environement.

WebSphere Application Server performance considerations

This web site provides resources for each version of WebSphere Application Server on the iSeries server, including many useful performance tips and recommendations. This resource is particularly valuable for environments using servlets, Java Server Pages (JSPs) and Enterprise Java Beans (EJBs).

DB2 UDB/WebSphere Performance Tuning Guide

This Redbook provides an introduction to both the WebSphere and DB2 environments, and offers suggestions, examples, and solutions to common performance problems that can help you optimize WebSphere and DB2 performance.

Java and WebSphere performance in OS/400

Use this Redbook to learn how to plan for and configure you operating environment to maximize Java and WebSphere performance, and to help you collect and analyze performance data.

WebSphere V3 Performance Tuning Guide

This Redbook offers detailed recommendations and examples for optimizing WebSphere V3 performance on the iSeries server.

iSeries Performance Capabilities Reference

The Performance Capabilities Reference provides detailed information, reports and examples that can help you configure or tune your iSeries server for optimal performance. In particular, see chapter 6: Web Server and Web Commerce, for WebSphere specific performance tips.

For other WebSphere and e-business information resources, refer to the topic WebSphere e-business administration.

Applications for performance management

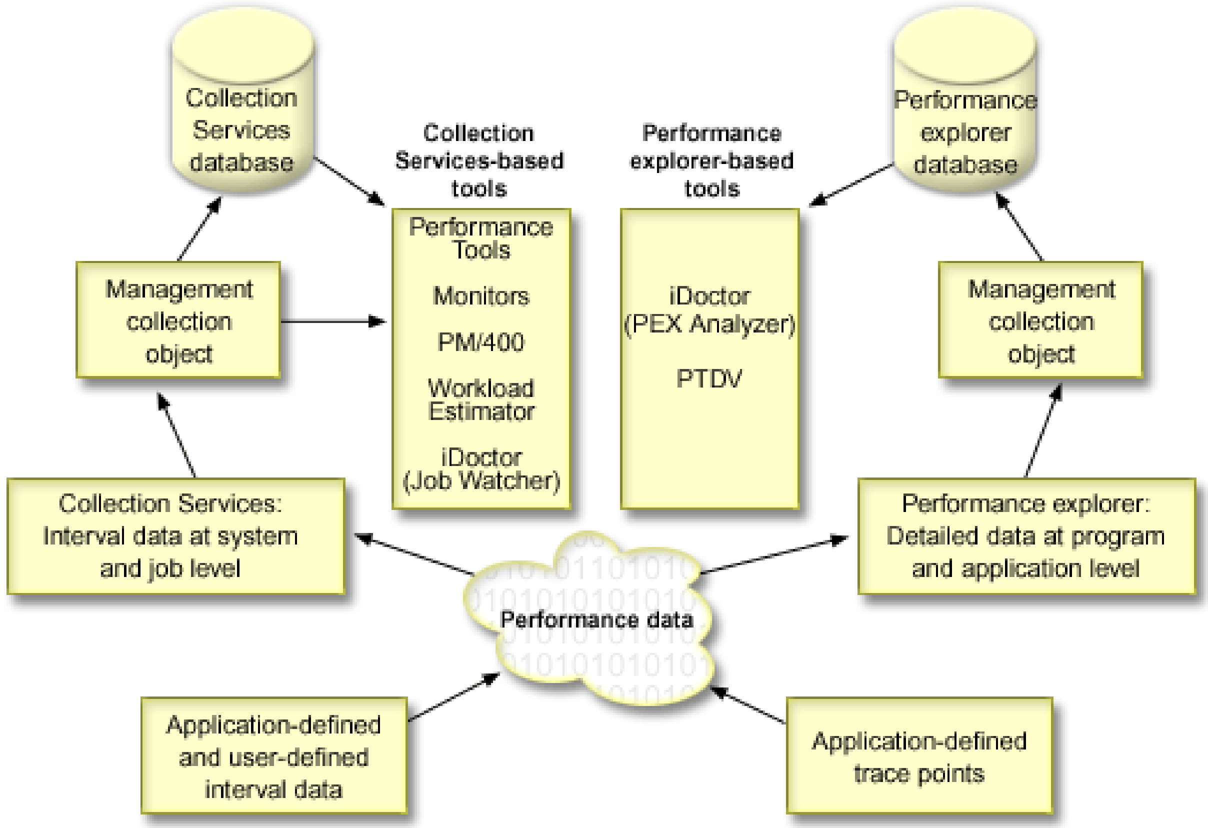
Many of the applications for performance management have several functions. Knowing exactly which component of the available suite of applications best suits a given situation can be complex. The following topics provide detailed information about each of the performance management applications, including selection, use, and configuration.

As shown in the following figure, basically there are two performance collection functions on the iSeries servers:

- Collection Services, which collects interval data at the system and job level. You can run this continuously to know what is happening with your system. The interval data that is collected is either application defined or user defined.
- Performance explorer, which collects detailed data at the program and application level. It also traces the flow of work in an application and can be used to diagnose difficult performance problems. The data that is collected is by application-defined performance explorer trace points, such as with Domino, NetServer, or WebSphere.

Both collection functions deposit their data into management collection objects. You can convert the data from the management collection objects by using the Create Performance Data (CRTPFRDTA) command for Collection Services data or by using the Create Performance Explorer Data (CRTPEXDTA) command for the performance explorer data.

This topic introduces the performance management applications that are available to work with either the Collection Services data or the performance explorer data.


Collection Services database
Collection Services-based tools
Performance explorer-based tools
Performance explorer database
Performance Tools
Monitors
PM/400
Workload Estimator
iDoctor (Job Watcher)
iDoctor (PEX Analyzer)
PTDV
Management collection object
Management collection object
Collection Services: Interval data at system and job level
Performance explorer: Detailed data at program and application level
Performance data
Application-defined and user-defined interval data
Application-defined trace points

Collection Services
Collection Services gathers performance data at user-defined time intervals and then stores this information in collection objects on your system. Many of the other tools, including monitors, graph history, PM/400, and many functions in the Performance Tools licensed program, rely on these collection objects for their data.

Performance database files
You can generate database files from the collection objects maintained by Collection Services. Use this topic to find the names, descriptions and attributes of these database files.

Monitors
Monitors display current information about the performance of your systems. Additionally, you can use them to carry out predefined actions when a specific event occurs. You can use the system, message, job, file, and B2B transaction monitors to display and monitor information about your systems. The system and job monitors use the performance data collected by Collection Services.

Graph history
Graph history provides a graphical display of performance data collected by Collection Services over a specified period of time.

PM/400
PM/400 automates the collection, archival, and analysis of system performance data and returns reports to help you manage system resources and capacity. PM/400 uses the performance data collected by Collection Services.

Performance Tools
The Performance Tools licensed program includes many features to help you gather, analyze, and maintain system performance information. This includes assistance in managing performance over a distributed network, collecting and reporting on both summary and trace data, and capacity planning. Performance Tools uses the performance data collected by Collection Services (sample data) and the trace data obtained from the Start Performance Trace (STRPFRTRC) command and the End Performance Trace (ENDPFRTRC) command.

Performance Explorer
Performance explorer collects more detailed information about a specific application, program or system resource, and provides detailed insight into a specific performance problem. This includes the capability both to perform several types and levels of traces and to run detailed reports.

iDoctor for iSeries
The iDoctor for iSeries plugin consists of three software tools for managing performance, Performance Explorer Analyzer for detailed trace data analysis, Job Watcher for trace-level information about a job's behavior, and Object Explorer to help you query and manage objects on your system.

Performance Trace Data Visualizer (PTDV)
Performance Trace Data Visualizer for iSeries (PTDV) is a Java application that can be used for performance analysis of applications running on iSeries.

Performance Management APIs
The Performance Management APIs provide services to manage collections. These APIs start, end, and cycle collections, and they change and retrieve system parameters for the data collected. Many of the Performance Management APIs use the performance data collected by Collection Services.

OS/400 performance commands
OS/400 includes several important functions to help you manage and maintain system performance.

Extended Adaptive Cache
Extended Adaptive Cache can improve system performance by collecting disk usage data, and then using those statistics to create a large cache, effectively reducing the physical I/O requests for the disk.

Workload Estimator for iSeries
Workload Estimator helps you plan the size and timing requirements of your next upgrade. This tool is often used with PM/400 to analyze trends in system performance and helps you efficiently manage the growth and expansion of your iSeries server.

iSeries Navigator for Wireless
iSeries Navigator for Wireless allows you to monitor performance data over a wireless connection, using a personal digital assistant (PDA), Internet-ready telephone, or traditional Web browser. The iSeries Navigator for Wireless uses the performance data collected by Collection Services.

PATROL for iSeries (AS/400) - Predict
PATROL for iSeries (AS/400) - Predict helps manage iSeries performance by automating many of the routine administration tasks required for high availability and optimal performance. Additionally, this product offers detailed capacity planning information to help you plan the growth of your iSeries environment.

Collection Services

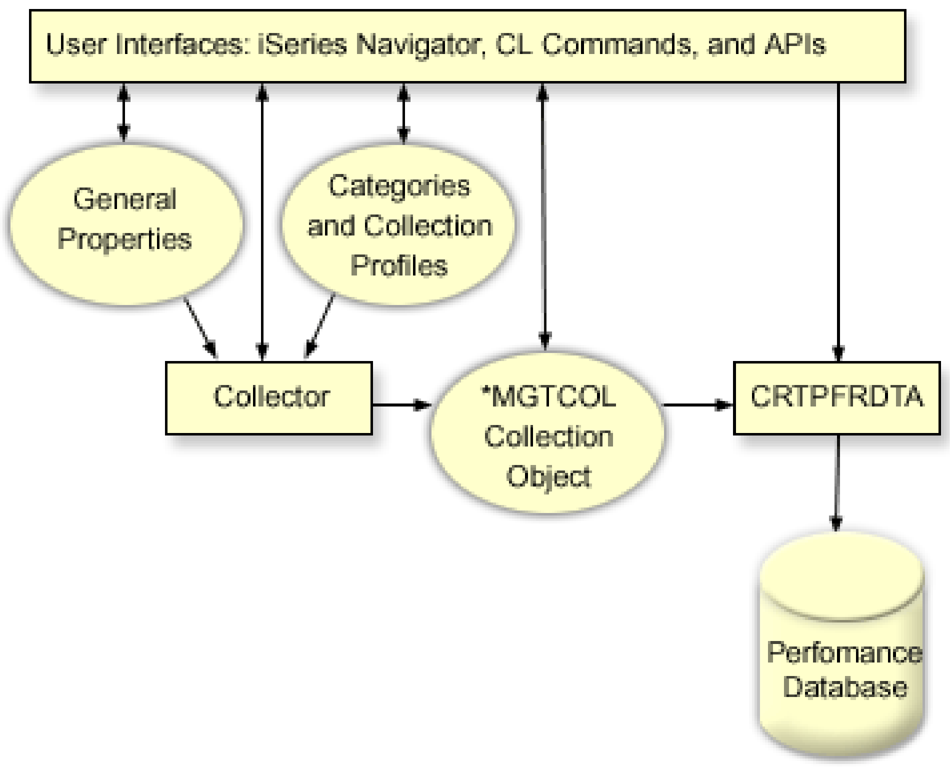
Use Collection Services to collect performance data for later analysis by the Performance Tools for iSeries licensed program or other performance report applications, iSeries Navigator monitors, and the graph history function. (If you prefer viewing real-time performance data, system monitors provide an easy-to-use graphical interface for monitoring system performance.) Collection Services collects data that identifies the relative amount of system resource used by different areas of your system. Use Collection Services to:

- Easily manage your collection objects
- Collect performance data continuously and automatically with minimal system overhead
- Control what data is collected and how the data is used
- Move performance data between releases without converting the data
- Create performance data files that are used by Performance Tools
- Integrate your own programs to collect user-defined performance data into Collection Services.

How Collection Services works

Collection Services replaces the OS/400 performance monitor, which was called by the Start Performance Monitor (STRPFRMON) command. The Performance Monitor (STRPFRMON command) has not been available since V4R5. When you used the OS/400 performance monitor, your data was collected into as many as 30 database files.

Collection Services capabilities introduce a new process of collecting performance data. Collection Services stores your data for each collection in a single collection object, from which you can create as many different sets of database files as you need. This means a lower system overhead when collecting performance data. Even if you elect to create the database files during collection, you still experience a performance advantage over the OS/400 performance monitor because Collection Services uses a lower priority (50) batch job to update these files. The reduction in collection overhead makes it practical to collect performance data in greater detail and at shorter intervals on a continuous basis. Collection Services enables you to establish a network-wide system policy for collecting and retaining performance data and to implement that policy automatically. For as long as you retain the management collection objects, if the need arises, you have the capability to look back and analyze performance-related events down to the level of detail that you collected.


User Interfaces: iSeries Navigator, CL Commands, and APIs
General Properties
Categories and Collection Profiles
Collector
*MGTCOL Collection Object
CRTPFRDTA
Perfomance Database

Collection Services allows you to gather performance data with little or no observable impact on system performance. You can use iSeries Navigator to configure Collection Services to collect the data you want as frequently as you want to gather it. A collection object, *MGTCOL, serves as an efficient storage medium to hold large quantities of performance data. Once you have configured and started Collection Services, performance data is continuously collected. When you need to work with performance data, you can copy the data you need into a set of performance database files.

The figure above provides an overview of the following Collection Services elements:

User interfaces
Several methods exist that allow you to access the different elements of Collection Services. For example, you can use CL commands, APIs, and the iSeries Navigator interface.

General properties
General properties define how a collection should be accomplished and they control automatic collection attributes.

Data categories
Data categories identify the types of data to collect. You can configure categories independently to control what data is collected and how often the data is collected.

Collection profiles
Collection profiles provide a means to save and activate a particular category configuration.

Performance collector
The performance collector uses the general properties and category information to control the collection of performance data. You can start and stop the performance collector, or configure it to run automatically.

Collection Object
The collection object, *MGTCOL, serves as an efficient storage medium for large quantities of performance data.

Create Performance Data (CRTPRFDTA) command
The CRTPFRDTA command processes data that is stored in the management collection object and generates the performance database files.

Performance database
The database files store the data that is processed by the CRTPFRDTA command. The files can be divided into these categories: Performance data files that contain time interval data, configuration data files, trace data files.

How to start Collection Services

You can start Collection Services by using any of the following methods. However, the information in the Performance topic focuses on iSeries Navigator methods.

Starting method	Description
iSeries Navigator	The section that follows below shows you how to do a variety of Collection Services tasks using iSeries Navigator.
Performance Management APIs	You can use Performance Management APIs to start, customize, end, and cycle collections. In addition, you can use the APIs to work with the management collection objects or define your own transactions.
Traditional menu options	Type **GO PERFORM** in the character-based interface and select option 2 (Collect performance data) from the Performance Tools main menu. For additional information, go to the Performance Tools book .
Performance Management/400	You can activate PM/400 which automates the start of Collection Services and then creates the database files during collection.

Collection Services tasks

You can use Collection Services and iSeries Navigator to perform a variety of data collection tasks as shown in the following table.

Task	Description
Start Collection Services in a variety of ways	Create a customized performance data collection on an individual system or groups of systems with specific performance metrics. You can also use a Start Collector API in your startup program to start performance data collections automatically. For more information about how to perform these tasks, refer to the online help. Detailed task help is available from the iSeries Navigator window. Just click **Help** from the menu bar and select **Help Topics**. Select **What can I do with . . .?** to find out what you can do and where you need to be in the iSeries Navigator window to make it happen.

Task	Description
Create database files	Use Collection Services to automate the creation of performance database files. You can also create database files from the collection object, where the data is stored after it has been collected. You can use these database files with PM/400 or with the Performance Tools licensed program, or you can create your own queries to run against these files. Learn how to control what data gets collected as you create database files. See Performance database files to find out what database files are available to you, as well as what field-level data is included in each file.
Customize data collections	Customize your data collections. Find information about controlling what performance data you collect and how often that data gets collected. You can also find information about important time zone considerations.
Add user-defined categories to Collection Services	You can collect performance data from user applications by writing an exit program and integrating it into Collection Services. You can then collect this data during routine collection intervals and store it in collection objects. For information on how to implement this, see User-defined categories.
Manage collection objects	Find the information you need to manage collection objects, including the contents of collection objects, how long collection objects are saved, and what you can do with collection objects.
Collect trace data	Collection Services collects sample data. However, it does not collect trace data. Find out how to collect trace data.
Collect performance data for user-defined transactions	Collection Services provides APIs that allow you to define your own transactions. Find out how to do this task with the User-defined transactions.

Create database files from Collection Services data

Collection Services places the data you collected into management collection objects. To use this data, you must first place the data in a special set of database files. To create database files automatically as data is collected, simply select **Create database files** on the **Start Collection Services** dialog. You can also create the database files later when you want to export data to them from an existing management collection object.

You have many options that allow you to create database files.

- When you use Collection Services to collect performance data, you can create database files automatically as data is collected.
- You can create database files from the management collection object, where the data is stored after it has been collected. You can use the Create Performance Data (CRTPFRDTA) command to create a set of performance database files from performance information stored in a management collection (*MGTCOL) object. You can use either the iSeries Navigator interface or the CRTPFRDTA command.
- You can activate Performance Management/400, which automates the start of Collection Services and then creates the database files during collection.

You can use the database files that you have created with the Performance Tools for iSeries licensed program or other applications to produce performance reports. You can collect the performance data on one system and then move the management collection object (*MGTCOL) to another system to generate the performance data files and run the Performance Tools reports. This action allows you to analyze the performance data on another system without affecting the performance of the source system. For more information about Performance Tools, see the Performance Tools book

.

Storing data in management collection objects instead of in database files

Why should you store the data in management collection objects instead of in the database files that you need to run your reports? Because you can manage the management collection objects separately from the database files, you can collect your performance data in small collection intervals (such as 5-minute intervals) and then create your database files with a longer sampling interval (such as 15-minute intervals).

From a single management collection object, you can create many different sets of database files for different purposes by specifying different data categories, different ranges of time, and different sampling intervals.

For example, you might collect performance data on the entire set of categories (all data, or the **Standard plus protocol** profile) in 5-minute collection intervals for 24 hours. From that one management collection object, you can create different sets of database files for different purposes. You could create one set of database files to run your normal daily performance reports. These files might contain data from all categories with a sampling interval of 15 minutes. Then, to analyze a particular performance problem, you could create another set of database files. These files might contain only data for a single category that you need to analyze, a specific time period within the 24 hours, and a more granular sampling interval of 5 minutes.

In addition, the single management collection object allows you to manage the data as a single object rather than as many files. The single collection object allows you to move the performance data between releases without converting the data. As long as you retain the collection objects, you can look back and analyze the performance-related events down to the level of detail that you collected.

Export the collected data

To export performance data from a management collection object to database files, follow these steps:
1. In iSeries Navigator, either select an endpoint system under **Management Central** or select a system to which you have a direct connection under **My Connections** (or your active environment).
2. Expand **Configuration and Service**.
3. Click **Collection Services**.
4. Right-click the management collection object that you want to export to database files and select **Create Database Files**.
5. On the **Create Database Files** dialog, select the categories from the collection object to include in the database files. You can also select a different time period and sampling interval, as long as the collection object contains data to support your selections.
6. Click **OK**.

Create database files from an existing collection object: You can export performance data from an existing management collection object to database files. Follow these steps:
1. Expand **Configuration and Service** for the system from which performance data is being collected.
2. Select **Collection Services**.
3. Right-click the management collection object from which you want to export data to the database files.
4. You can first select **Properties** to display the characteristics of the data in the collection object. On the Data properties page, you can see the categories of data collected in this collection object as well as the intervals at which they were collected. You can use this information in selecting the data that you want to export. When you have reviewed this information, click **OK**.
5. Right-click the management collection object again and select **Create Database Files**. Complete the fields using the online help.
6. Click **OK**.

After you convert the data in the database files, you can use the Performance Tools for iSeries licensed program or other applications to produce performance reports.

Customize data collections

When you use Collection Services to collect performance data, you control what data is collected and how often it is collected. You can select from the collection profiles that are provided. The **Standard** profile corresponds to the settings for system data in the OS/400 performance monitor function that was provided by the Start Performance Monitor (STRPFRMON) command in previous releases. The **Standard plus protocol** profile corresponds to the STRPFRMON command settings for all data. Or you can select **Custom** to create your own customized profile. There are also several other profiles available, refer to the online help for detailed descriptions. For your customized profile, you can select from a list of available data categories, such as System CPU, Local Response Time, Disk Storage, and IOPs (input/output processors).

For each category of data that you collect, you can specify how often the data will be collected. For many categories, you will want to select the default collection interval, which you can set from predefined settings between 15 seconds and 60 minutes. (The recommended setting is 15 minutes.)

Note: When the default value is set to any specified time, all categories except those categories with explicit time intervals, such as, disk storage, input/output processors, and communications-related categories, use the specified time.

The collected data is stored in a management collection object (type *MGTCOL) called a collection. To prevent these management collection objects from becoming too large, the collection must be cycled at regular intervals. Cycling a collection means to create a new collection object and begin storing data in it at the same time data collection stops in the original collection object. You can specify any interval from one hour to 24 hours, depending on how you plan to use the data.

To customize Collection Services on a system, follow these steps:

1. In iSeries Navigator, select either an endpoint system under **Management Central** or a system to which you have a direct connection under **My Connections** (or your active environment).
2. Expand **Configuration and Service**.
3. Right-click **Collection Services** and select **Properties**.
4. On the **General** page, you may want to specify a retention period longer than the default of 1 day. Collection Services may delete management collection objects and the data they contain from the system at any time after the retention period has expired. When the management collection object is created, an expiration date is assigned to it. Even if you move the collection object to another library, Collection Services will delete the object after it expires. You can specify **Permanent** if you do not want Collection Services to assign an expiration date to new collection objects. You will then have to delete these collection objects manually.
   To view the Graph History window, you must specify a Collection retention period of either Graph or Summary. When you specify these options, you can take advantage of the historical reporting capabilities, which allow you to do metric comparisons for multiple systems over extended periods of time.
   You can also specify the path of the location where you want to store your collections, how often you want to cycle collections, and the default collection interval. You can select to create database files automatically during collection.
5. Click the **Data to Collect** tab.
6. For **Collection profile to use**, select **Custom**. You can specify the collection interval for each category you select for your customized list.
7. Click **OK** to save your customized values.

Once you have customized Collection Services to the settings you prefer, you can right-click **Collection Services** again and select **Start Collection Services** to begin collecting performance data.

Time zone considerations for Collection Services

When you review and analyze performance data, the actual local time of the collection can be significant. For example, you may need to be sure which data was collected during the busiest period of the day so that it represents the heaviest workload experienced by the system under review. If some of the systems from which you collect performance data are located in different time zones, you should be aware of these considerations:

- When you start Collection Services for a system group, you start Collection Services at the same time on all systems in the group. Any differences in system time and date settings due to some systems being located in different time zones are not taken into account.
- If you start Collection Services with the Management Central scheduler, the time at which the scheduler starts the task is based on the system time and date of your central system in Management Central.
- The management collection objects for each endpoint system reflect start and end times based on the QTIME and QUTCOFFSET (coordinated universal time offset) system values of that endpoint system and your central system. If the endpoint system is in a different time zone from your central system, and these system values are correctly set on both systems, the start and end times reported for collection objects are the actual times on the endpoint system. In other words, the start and end times reflect the value of QTIME on the endpoint system as it was at the actual point in time when those events occurred.
- The scheduling of a performance collection can cross a boundary from standard time to daylight savings time or from daylight savings time to standard time. If so, this time difference should be taken into account when scheduling the start time. Otherwise, the actual start and end times can be an hour later or earlier than expected. In addition, the reported start and end times for management collection objects are affected by this difference unless the QUTCOFFSET system value is adjusted each time the change to and from daylight savings time takes effect.

For more information about using Collection Services to collect performance data, see Collection Services.

User-defined categories in Collection Services

The user-defined categories function in Collection Services enables applications to integrate performance data collection into Collection Services. This allows you to gather data from an application by writing a data collection program, registering it, and integrating it with Collection Services. Collection Services will then call the data collection program at every collection interval, and will store the data in the collection object. You should use the Collection Object APIs listed below to access the data stored in the collection object. You may access the data in real-time, as it is being collected, or for as long as the collection object is retained.

To implement this function, you need to:

1. Develop a program to collect performance data for a new category in Collection Services. Refer to Collection program recommendations and requirements for more information.
2. Create a job description for your collection program. The job description QPMUSRCAT in QGPL provides an example, but does not represent default values or recommendations.
3. Register the new category and specify the data collection program. See the API descriptions for more information:
   - Register: QypsRegCollectorDataCategory
   - De-register: QypsDeregCollectorDataCategory

   After you register the category, Collection Services includes it in the list of available collection categories.
4. Add the category to your Collection Services profile, and then cycle Collection Services
5. Develop a program to query the collection object. See the API descriptions for more information:
   - Retrieve active management collection object name: QpmRtvActiveMgtcolName (Used only for querying the collection object in real-time.)
   - Retrieve management collection object attributes: QpmRtvMgtcolAttrs

- Open management collection object: QpmOpenMgtcol
- Close management collection object: QpmCloseMgtcol
- Open management collection object repository: QpmOpenMgtcolRepo
- Close management collection object repository: QpmCloseMgtcolRepo
- Read management collection object data: QpmReadMgtcolData

Your customized collection program now runs at each collection interval, and the collected data is archived in the collection objects.

You can also implement the Java versions of these APIs. The required Java classes are included in ColSrv.jar, in the IFS directory QIBM/ProdData/OS400/CollectionServices/lib. Java applications should include this file in their classpath. For more information about the Java implementation, refer to the javadocs.

For an example implementation, see Example: Implementing user-defined categories

Query the collection object in real-time

If your application needs to query the collection object in real-time, it will need to synchronize the queries with Collection Services. To do this, the application should create a data queue and register it with Collection Services. Once registered, the collector sends a notification for each collection interval and for the end of the collection cycle. The application should maintain the data queue, including removing the data queue when finished, and handling abnormal termination. To register and de-register the data queue, refer to the following API descriptions:
- Add collector notification: QypsAddCollectorNotification
- Remove collector notification: QypsRmvCollectorNotification

Collection program recommendations and requirements: Collection Services calls the data collection program once during the start of a collection cycle, once for each collection interval, and again at the end of the collection cycle. The data collection program must perform any data collection and return that data to a data buffer provided by Collection Services. In addition to providing a data buffer, Collection Services also provides a work area, which allows the data collection program to maintain some state information between collection intervals.

The data collection program should collect data as quickly as possible and should perform minimal formatting. The program should not perform any data processing or sorting. Although data from the user-defined category is not converted into database files, Collection Services may run the CRTPFRDTA command automatically and add the data from the collection object to database files at the end of each collection interval. If the data collection program cannot complete its tasks within the collection interval, the CRTPFRDTA command does not run properly.

You may create the data collection program in several environments:
- *PGM for OPM languages. This environment may not be used to query the collection object and may result in poor performance. However, it is supported for older programming languages.
- *SRVPGM, an entry point in a service program. This is for ILE languages.
- *JVAPGM, the required Java classes are included in ColSrv.jar. This file is included in the IFS at QIBM/ProdData/OS400/CollectionServices/lib. Download the javadocs .zip file and open index.html for a description of the Java implementations of the APIs.

Collection Services sends the following requests to the data collection program:

Request	Description
Start collection	The data collection program should initialize any interfaces or resources used during data collection. Optionally, it may also initialize a work area, provided by Collection Services, that preserves state information between collection intervals. If you want to include a control record prior to the collected data, the data collection program may also write a small amount of data to the data buffer. Typically, this control record would be used during data processing to help interpret the data.
Collection interval	Collection Services sends an interval request for each collection interval. The data collection program should collect data and return it in the data buffer. Collection Services then writes that data to the interval record in the collection object. If the amount of data is too large for the data buffer, the data collection program should set a "More data" flag. This action causes Collection Services to send another interval request with a modifier indicating that it is a continuation. Collection Services resets the more data flag before each call. This process is repeated until all the data is moved into the collection object.
End of collection	When the collection for the category containing the data collection program ends, Collection Services sends this request. The data collection program should perform any cleanup and can optionally return a collection control record. The data collection program should also send a return code that indicates the result of the collection.
Clean up and terminate (Shutdown)	Collection Services sends this request if an abnormal termination is necessary. Operating system resources are freed automatically when the data collection program job ends, but any other shutdown operations should be performed by the data collection program. The data collection program can receive this request at any time.

For a detailed description of these parameters, the work area, data buffer, and return codes, refer to the header file QPMDCPRM, which is located in QSYSINC.

Data storage in collection objects

Collection objects have a repository for each data collection category. This repository gets created by Collection Services when collections for that category are started. Each repository consists of the following records:

Record	Description
Control	This optional record can be the first or last record that results from the data collection program, and may occur in both positions. Typically, it should contain any information needed to interpret the record data.
Interval	Each collection interval creates an interval record, even if it is empty. The interval record contains the data written to the data buffer during the collection interval. It must not exceed 4 GB in size.
Stop	Collection Services automatically creates this record to indicate the end of a data collection session. If the collections for the user-defined category were restarted without ending or cycling Collection Services, you can optionally include a control record followed by additional interval records after the stop record.

Example: Implementing user-defined categories: The following sample programs illustrate how you can use the provided APIs to integrate customized data collections into Collection Services.

- Sample data collection program (C++)
- Sample program to register the data collection program (C++)
- Sample program to query the collection object (Java)

Code example disclaimer

Example: data collection program: The following program collects some test data and stores it in a data buffer, which Collection Services copy to the collection object. For more information about the collection program, refer to Collection program recommendations and requirements.

Note: Read the Code example disclaimer for important legal information.

C++ sample code

```
#include "string.h"                        // memcpy(), memset(), strlen()
#include "stdio.h"                        // printf()
#include "qpmdcprm.h"                      // data collection program interface
#include "time.h"

extern "C"
void DCPentry( Qpm_DC_Parm_t *request, char *dataBuffer,
                                                    char *workArea, int *returnCode )
{
  static  char  testData[21] = "Just some test stuff";
  int           i;



/* Print contents of request structure */

  printf( "DCP called with parameters:\n" );
  printf( "  format name: \"%8.8s\";  category name: \"%10.10s\";\n",
          request->formatName, request->categoryName );
  printf( "  rsvd1: %4.4X; req type: %d; req mod: %d; buffer len: %d;\n",
          *(short *)(request->rsvd1), request->requestType,
          request->requestModifier, request->dataBufferLength );
  printf( "  prm offset: %d; prm len: %d; work len: %d; rsvd2: %8.8X;\n",
          request->parmOffset, request->parmLength, request->workAreaLength,
          *(int *)(request->rsvd2) );
  printf( "  rec key: \"%8.8s\"; timestamp: %8.8X %8.8X;\n",
          request->intervalKey,
          *(int *)(request->intervalTimestamp),
          *(int *)(request->intervalTimestamp + 4) );
  printf( "  return len: %d; more data: %d; rsvd3: %8.8X %8.8X;\n",
          request->bytesProvided, request->moreData,
          *(int *)(request->rsvd3),
          *(int *)(request->rsvd3 + 4) );

  switch ( request->requestType )
  {
  /* Write control record in the beginning of collection */
    case PM_DOBEGIN:
      printf( "doBegin(%d)\n", request->requestModifier );
      switch ( request->requestModifier)
      {
        case PM_CALL_NORMAL:
            memcpy( dataBuffer, testData, 20 );
            *(int *)workArea = 20;
            request->moreData = PM_MORE_DATA;
            request->bytesProvided = 20;
          break;

        case PM_CALL_CONTINUE:
          if ( *(int *)workArea < 200 )
          {
            memcpy( dataBuffer, testData, 20 );
            *(int *)workArea += 20;
            request->moreData = PM_MORE_DATA;
            request->bytesProvided = 20;
          }
          else
          {
            *(int *)workArea = 0;
            request->moreData = PM_NO_MORE_DATA;
            request->bytesProvided = 0;
          }
          break;

        default:
          *returnCode = -1;
          return;
      }
      break;
/* Write control record in the end of collection */
   case PM_DOEND:
      printf( "doEnd(%d)\n", request->requestModifier );
      switch ( request->requestModifier)
      {
        case PM_CALL_NORMAL:
            memcpy( dataBuffer, testData, 20 );
            *(int *)workArea = 20;
            request->moreData = PM_MORE_DATA;
            request->bytesProvided = 20;
          break;

        case PM_CALL_CONTINUE:
          if ( *(int *)workArea < 200 )
          {
            memcpy( dataBuffer, testData, 20 );
            *(int *)workArea += 20;
            request->moreData = PM_MORE_DATA;
            request->bytesProvided = 20;
          }
          else
          {
            *(int *)workArea = 0;
            request->moreData = PM_NO_MORE_DATA;
            request->bytesProvided = 0;
          }
          break;

        default:
          *returnCode = -1;
          return;
      }
      break;

/*Write interval record */
   case PM_DOCOLLECT:
      printf( "doCollect(%d)\n", request->requestModifier );
      for ( i = 0; i < 10000; i++ )
        dataBuffer[i] = i % 256;
      request->bytesProvided = 10000;

      switch ( request->requestModifier)
      {
        case PM_CALL_NORMAL:
            *(time_t *)(workArea + 4) = time( NULL );
            *(int *)workArea = 1;
            request->moreData = PM_MORE_DATA;
          break;

        case PM_CALL_CONTINUE:
          *(int *)workArea += 1;
          if ( *(int *)workArea < 20 )
            request->moreData = PM_MORE_DATA;
          else
          {
            *(time_t *)(workArea + 8) = time( NULL );
            printf( "doCollect() complete in %d secs (%d bytes transferred)\n",
                    *(time_t *)(workArea + 8) - *(time_t *)(workArea + 4), 10000 * 20 );
            request->moreData = PM_NO_MORE_DATA;
          }
          break;

        default:
          *returnCode = -1;
          return;
      }
      break;
 /* Clean-up and terminate */
    case PM_DOSHUTDOWN:
      printf( "doShutdown\n" );
      *returnCode = 0;
      return;
      break;

    default:
      *returnCode = -1;
      return;
      break;
  }

}/* DCPentry() */
```

Example: Program to register the data collection program: The following program registers the data collection program from the previous example with Collection Services. After running, Collection Services displays the data collection program in the list of data collection categories.

C++ sample code

```
#include "stdlib.h"
#include "stdio.h"
#include "string.h"
#include "qypscoll.cleinc"


int main( int argc, char *argv[] )
{
    int    CCSID = 0;
    int    RC  = 0;
    Qyps_USER_CAT_PROGRAM_ATTR    *pgmAttr;
    Qyps_USER_CAT_ATTR             catAttr;
    char   collectorName[11] = "*PFR      ";
    char   categoryName[11]  = "TESTCAT   ";
    char   collectorDefn[11] = "*CUSTOM   ";  /* Register to *CUSTOM profile only */

      if ( argc > 2 )
      {
        int len = strlen( argv[2] );

        if ( len > 10 ) len = 10;
        memset( categoryName, ' ', 10 );
        memcpy( categoryName, argv[2], len );
      }

      if ( argc < 2 || *argv[1] == 'R' )
      {
        pgmAttr = (Qyps_USER_CAT_PROGRAM_ATTR *)malloc( 4096 );
        memset( pgmAttr, 0x00, sizeof(pgmAttr) );
        pgmAttr->fixedPortionSize = sizeof( Qyps_USER_CAT_PROGRAM_ATTR );
        memcpy( pgmAttr->programType,     "*SRVPGM   ", 10 );
        memcpy( pgmAttr->parameterFormat, "PMDC0100", 8 );
        memcpy( pgmAttr->ownerUserId,     "USERID    ", 10 );
        memcpy( pgmAttr->jobDescription,  "QPMUSRCAT QGPL      ", 20 );
        memcpy( pgmAttr->qualPgmSrvpgmName, "DCPTEST   LIBRARY    ", 20 );
        pgmAttr->workAreaSize = 123;
        pgmAttr->srvpgmEntrypointOffset = pgmAttr->fixedPortionSize;
        pgmAttr->srvpgmEntrypointLength = 8;
        pgmAttr->categoryParameterOffset = pgmAttr->srvpgmEntrypointOffset +
                                           pgmAttr->srvpgmEntrypointLength;
        pgmAttr->categoryParameterLength = 10;
   /* Set entry point name */
        memcpy( (char *)(pgmAttr) + pgmAttr->srvpgmEntrypointOffset,
                "DCPentry", pgmAttr->srvpgmEntrypointLength );  /* Set parameter string */
        memcpy( (char *)(pgmAttr) + pgmAttr->categoryParameterOffset,
                "1234567890", pgmAttr->categoryParameterLength );

        memset( &catAttr, 0x00, sizeof(catAttr) );
        catAttr.structureSize = sizeof( Qyps_USER_CAT_ATTR );
        catAttr.minCollectionInterval = 0;
        catAttr.maxCollectionInterval = 0;
        catAttr.defaultCollectionInterval = 30;   /* Collect at 30 second interval */
        memset( catAttr.qualifiedMsgId, ' ', sizeof(catAttr.qualifiedMsgId) );
        memcpy( catAttr.categoryDesc,
                "12345678901234567890123456789012345678901234567890", sizeof(catAttr.categoryDesc) );

        QypsRegCollectorDataCategory( collectorName,
                                      categoryName,
                                      collectorDefn,
                                      &CCSID,
                                      (char*)pgmAttr,
                                      (char*)&catAttr,
                                      &RC
                                      );
      }
      else
      if( argc >= 2 && *argv[1] == 'D' )
        QypsDeregCollectorDataCategory( collectorName, categoryName, &RC );
      else
        printf("Unrecognized option\n");

}/* main() */
```

Example: Program to query the collection object: The following sample program illustrates how to query the data stored in the collection object using the Java classes shipped in the ColSrv.jar file in QIBM/ProdData/OS400/CollectionServices/lib.

Java sample code

```
import com.ibm.iseries.collectionservices.*;

class testmco2
{
  public static void main( String argv[] )
  {
    String    objectName = null;
    String    libraryName = null;
    String    repoName = null;
    MgtcolObj mco = null;
    int       repoHandle = 0;
    int       argc = argv.length;
    MgtcolObjAttributes
              attr = null;
    MgtcolObjRepositoryEntry
              repoE = null;
    MgtcolObjCollectionEntry
              collE = null;
    int       i,j;

    if ( argc <  3 )
    {
      System.out.println("testmco2  objectName libraryName repoName");
      System.exit(1);
    }

    objectName  = argv[0];
    libraryName = argv[1];
    repoName    = argv[2];

    if ( ! objectName.equals( "*ACTIVE" ) )
      mco = new MgtcolObj( objectName, libraryName );
    else
      try
      {
        mco = MgtcolObj.rtvActive();
      } catch ( Exception e)
      {
        System.out.println("rtvActive(): Exception " + e );
        System.exit(1);
      }
    System.out.println("Object name = " + mco.getName() );
    System.out.println("Library name = " + mco.getLibrary() );

    try
    {
      attr = mco.rtvAttributes( "MCOA0100" );
    } catch ( Exception e)
    {
      System.out.println("rtvAttributes(): MCOA0100: Exception " + e );
      System.exit(1);
    }

    System.out.println("MCOA0100:  Object " + mco.getLibrary() + "/" + mco.getName() );
    System.out.println("   size = " + attr.size + " retention = " + attr.retentionPeriod +
         " interval = " + attr.dftInterval + " time created = " + attr.timeCreated +
         " time updated = " + attr.timeUpdated );
    System.out.println("   serial = " + attr.logicalPSN + " active = " + attr.isActive +
         " repaired = " + attr.isRepaired + " summary = " + attr.sumStatus +
         " repo count = " + attr.repositoryCount );
    if ( attr.repositoryInfo != null )
      for(i = 0; i < attr.repositoryCount; i++ )
      {
 repoE = attr.repositoryInfo[ i ];
 System.out.println("      name = " + repoE.name + " category = " + repoE.categoryName +
      " size = " + repoE.size );
 for( j = 0; j < repoE.collectionInfo.length; j++ )
 {
   collE = repoE.collectionInfo[ j ];
   System.out.println("        startTime = " + collE.startTime + " endTime = " + collE.endTime +
        " interval = " + collE.interval );
 }
      }

    try
    {
      attr = mco.rtvAttributes( "MCOA0200" );
    } catch ( Exception e)
    {
      System.out.println("rtvAttributes(): MCOA0200: Exception " + e );
      System.exit(1);
    }

    System.out.println("MCOA0200: Object " + mco.getLibrary() + "/" + mco.getName() );
    System.out.println("   size = " + attr.size + " retention = " + attr.retentionPeriod +
         " interval = " + attr.dftInterval + " time created = " + attr.timeCreated +
         " time updated = " + attr.timeUpdated );
    System.out.println("   serial = " + attr.logicalPSN + " active = " + attr.isActive +
         " repaired = " + attr.isRepaired + " summary = " + attr.sumStatus +
         " repo count = " + attr.repositoryCount );
    if ( attr.repositoryInfo != null )
      for(i = 0; i < attr.repositoryCount; i++ )
      {
 repoE = attr.repositoryInfo[ i ];
 System.out.println("      name = " + repoE.name + " category = " + repoE.categoryName +
      " size = " + repoE.size );
 for( j = 0; j < repoE.collectionInfo.length; j++ )
 {
   collE = repoE.collectionInfo[ j ];
   System.out.println("         startTime = " + collE.startTime + " endTime = " + collE.endTime +
        " interval = " + collE.interval );
 }
      }

    if ( repoName.equals("NONE") )
 return;

    try
    {
      mco.open();
    } catch ( Exception e)
    {
      System.out.println("open(): Exception " + e );
      System.exit(1);
    }

    try
    {
      repoHandle = mco.openRepository( repoName, "MCOD0100" );
    } catch ( Exception e)
    {
      System.out.println("openRepository(): Exception " + e );
      mco.close();
      System.exit(1);
    }
    System.out.println("repoHandle = " + repoHandle );

    MgtcolObjReadOptions  readOptions = new MgtcolObjReadOptions();
    MgtcolObjRecInfo recInfo = new MgtcolObjRecInfo();

    readOptions.option = MgtcolObjReadOptions.READ_NEXT;
    readOptions.recKey = null;
    readOptions.offset = 0;
    readOptions.length = 0;

    while ( recInfo.recStatus == MgtcolObjRecInfo.RECORD_OK )
    {
      try
      {
        mco.readData( repoHandle, readOptions, recInfo, null );
      } catch ( Exception e)
      {
        System.out.println("readData(): Exception " + e );
        mco.close();
        System.exit(1);
      }

      if( recInfo.recStatus == MgtcolObjRecInfo.RECORD_OK )
      {
        System.out.print("Type = " + recInfo.recType );
        System.out.print(" Key = " + recInfo.recKey );
        System.out.println(" Length = " + recInfo.recLength );
      }

    }/* while ... */

    mco.closeRepository( repoHandle );
    mco.close();

  }/* main() */

}/* class testmco2 */
```

Manage collection objects

When you use Collection Services to collect performance data, each collection is stored in a single object. You can see a summary of the data in any management collection object by following these steps:

1. In iSeries Navigator, select either an endpoint system under **Management Central** or a system to which you have a direct connection under **My Connections** (or your active environment).
2. Expand **Configuration and Service**.
3. Select **Collection Services**.
4. Right-click any management collection object in the list and select **Properties** to see general information about that collection and a summary of the data that it contains.

You can right-click any collection object and select Create database files to specify the data categories, the range of time within the collection period, and the sampling interval that you want to include in the database files.

You can right-click any collection object and select Graph History to graphically view the data in the management collection object.

Delete or keep old management collection objects

You can delete a collection object from the system by right-clicking the object and selecting **Delete**. If you do not delete the objects manually, Collection Services will delete them automatically after the expiration date and time.

Collection Services deletes only **cycled** management collection objects. A status of **Cycled** means that Collection Services has stopped collecting data and storing it in the object. The status of each management collection object is shown in the list of collection objects when you expand **Configuration and Service** and select **Collection Services.**

Collection Services deletes the cycled collection objects that have reached their expiration date and time the next time it starts or cycles a collection. The expiration date is associated with the management collection object. Even if you move the collection object to another library, Collection Services will delete the object after it expires.

The expiration date for each management collection object is shown in the Properties for that collection object. To keep the object on the system longer, you simply change the date on the Properties page. Right-click any management collection object in the list and select **Properties** to see the information about that collection. You can specify **Permanent** if you do not want Collection Services to delete your management collection objects for you.

User-defined transactions

Collection Services and performance explorer can now collect performance data that you define in your applications. With the provided APIs, you can integrate transaction data into the regularly scheduled sample data collections using Collection Services, and get trace-level data about your transaction by running performance explorer.

For detailed descriptions and usage notes, refer to the following API descriptions:

- Start transaction: QYPESTRT, qypeStartTransaction
- End transaction: QYPEENDT, qypeEndTransaction
- Log transaction: QYPELOGT, qypeLogTransaction (Used only by performance explorer)
- Add trace point: QYPEADDT, qypeAddTracePoint (Used only by performance explorer)

Note: You only need to instrument your application once. Collection Services and Performance Explorer use the same API calls to gather different types of performance data.

Integrating user-defined transaction data into Collection Services

You can select user-defined transactions as a category for collection in the Collection Services configuration. Collection Services then collects the transaction data at every collection interval and stores that data in the collection object. The CRTPFRDTA command exports this data to the user-defined transaction performance database file, QAPMUSRTNS. Collection Services organizes the data by transaction type. You can specify as many transaction types as you require; however, Collection services will only report the first fifteen transaction types. Data for additional transaction types is combined and stored as a *OTHER transaction type. At every collection interval, Collection Services creates one record for each type of transaction for each unique job. For a detailed description, refer to the usage notes in the Start transaction API.

Collection Services gathers general transaction data, such as the transaction response time. You can also include up to 16 optional application-defined counters that can track application specific data like the number of SQL statements used for the transaction, or other incremental measurements. Your application should use the Start transaction API to indicate the beginning of a new transaction, and should include a corresponding End transaction API to deliver the transaction data to Collection Services. For more information, refer to the QAMUSRTNS file description and the API descriptions.

For a sample implementation, refer to the examples in C++ or Java.

Note: Read the Code example disclaimer for important legal information.

Collecting trace information for user-defined transactions with performance explorer

You can use the start, end, and log transaction APIs during a performance explorer session to create a trace record. Performance Explorer stores system resource utilization, such as CPU utilization, I/O and seize/lock activity, for the current thread in these trace records. Additionally, you may choose to include application-specific performance data, and then send it to performance explorer in each of these APIs. You can also use the add trace point API to identify application-specific events for which performance explorer should collect trace data.

To start a performance explorer session for your transactions, specify *USRTRNS on the (OSEVT) parameter of your Performance Explorer definition. After entering the ENDPEX command, performance explorer writes the data supplied by the application to the QMUDTA field in the QAYPEMIUSR performance explorer database file. System-supplied performance data for the start, end, and any log records is stored in the QAYPEMIUSR and QAYPETIDX database files.

For a detailed description, refer to the API descriptions and the usage notes in the Start transaction API description.

C++ example: Integrating user-defined transactions into Collection Services: The following C++ example program shows how to use the Start transaction and End transaction APIs to integrate user-defined transaction performance data into Collection Services.

Note: Read the Code example disclaimer for important legal information.

```
//*********************************************************************
// tnstst.C
//
// This example program illustrates the use
// of the Start/End Transaction APIs (qypeStartTransaction,
// qypeEndTransaction).
//
//
// This program can be invoked as follows:
//   CALL lib/TNSTST PARM('threads' 'types' 'transactions' 'delay')
//     where
//       threads      = number of threads to create (10000 max)
//       types        = number of transaction types for each thread
//       transactions = number of transactions for each transaction
//                      type
//       delay        = delay time (millisecs) between starting and
//                      ending the transaction
//
// This program will create "threads" number of threads. Each thread
// will generate transactions in the same way. A thread will do
// "transactions" number of transactions for each transaction type,
// where a transaction is defined as a call to Start Transaction API,
// then a delay of  "delay" millisecs, then a call to End Transaction
// API. Thus, each thread will do a total of "transactions" * "types"
// number of transactions. Each transaction type will be named
// "TRANSACTION_TYPE_nnn" where nnn ranges from 001 to "types". For
// transaction type n, there will be  n-1 (16 max) user-provided
// counters reported, with counter m reporting m counts for each
// transaction.
//
// This program must be run in a job that allows multiple threads
// (interactive jobs typically do not allow multiple threads). One
// way to do this is to invoke the program using the SBMJOB command
// specifying ALWMLTTHD(*YES).
//
//*********************************************************************

#define _MULTI_THREADED

// Includes
#include "pthread.h"
#include "stdio.h"
#include "stdlib.h"
#include "string.h"
#include "qusec.h"
#include "lbcpynv.h"
#include "qypesvpg.h"

// Constants
#define maxThreads 10000

// Transaction pgm parm structure
typedef struct
{
  int types;
  int trans;
  int delay;
```

```
} tnsPgmParm_t;

// Error code structure
typedef struct
{
  Qus_EC_t error;
  char     Exception_Data[100];
} error_code_t;


//*********************************************************************
//
// Transaction program to run in each secondary thread
//
//*********************************************************************

void *tnsPgm(void *parm)
{
  tnsPgmParm_t *p = (tnsPgmParm_t *)parm;

  char tnsTyp[] = "TRANSACTION_TYPE_XXX";
  char pexData[] = "PEX";
  unsigned int pexDataL = sizeof(pexData) - 1;
  unsigned long long colSrvData[16] = {1,2,3,4,5,6,7,8,
                                       9,10,11,12,13,14,15,16};
  unsigned int colSrvDataL;
  char tnsStrTim[8];

  struct timespec ts = {0, 0};

  error_code_t errCode;

  _DPA_Template_T target, source; // Used for LBCPYNV MI instr

  unsigned int typCnt;
  unsigned int tnsCnt;
  int rc;


  // Initialize error code
  memset(&errCode, 0, sizeof(errCode));
  errCode.error.Bytes_Provided = sizeof(errCode);

  // Initialize delay time
  ts.tv_sec  = p->delay / 1000;
  ts.tv_nsec = (p->delay % 1000) * 1000000;

  // Loop doing transactions
  for (tnsCnt = 1; tnsCnt <= p->trans; tnsCnt++)
  {
    for (typCnt = 1; typCnt <= p->types; typCnt++)
    {
      // Set number field in transaction type
      source.Type = _T_UNSIGNED;
      source.Length = 4;
      source.reserved = 0;
      target.Type = _T_ZONED;
      target.Length = 3;
      target.reserved = 0;
      _LBCPYNV(tnsTyp + 17, &target, &typCnt, &source);

      // Set Coll Svcs data length in bytes
      colSrvDataL = (typCnt <= 16) ? (typCnt - 1) : 16;
      colSrvDataL = colSrvDataL * 8;

      // Call Start Transaction API
      qypeStartTransaction(tnsTyp,
                           (unsigned int *)&tnsCnt,
                           pexData,
                           (unsigned int *)&pexDataL,
                           tnsStrTim,
                           &errCode);

      // Delay specified amount
      rc = pthread_delay_np(&ts);

      // Call End Transaction API
      qypeEndTransaction(tnsTyp,
                         (unsigned int *)&tnsCnt,
                         pexData,
                         (unsigned int *)&pexDataL,
                         tnsStrTim,
                         (unsigned long long *)&colSrvData[0],
                         (unsigned int *)&colSrvDataL,
                         &errCode);
    }
  }

  return NULL;
}


//**********************************************************************
//
// Main program to run in primary thread
//
//**********************************************************************

void main(int argc, char *argv[])
{
  // Integer version of parms
  int threads;  // # of threads
  int types;    // # of types
  int trans;    // # of transactions
  int delay;    // Delay in millisecs

  pthread_t threadHandle[maxThreads];
  tnsPgmParm_t tnsPgmParm;
  int rc;
  int i;


  // Verify 4 parms passed
  if (argc != 5)
  {
    printf("Did not pass 4 parms\n");
    return;
  }

  // Copy parms into integer variables
  threads = atoi(argv[1]);
  types   = atoi(argv[2]);
  trans   = atoi(argv[3]);
  delay   = atoi(argv[4]);

  // Verify parms
  if (threads > maxThreads)
  {
    printf("Too many threads requested\n");
    return;
  }

  // Initialize transaction pgm parms (do not modify
  // these while threads are running)
  tnsPgmParm.types = types;
  tnsPgmParm.trans = trans;
  tnsPgmParm.delay = delay;

  // Create threads that will run transaction pgm
  for (i=0; i < threads; i++)
  {
    // Clear thread handle
    memset(&threadHandle[i], 0, sizeof(pthread_t));
    // Create thread
    rc = pthread_create(&threadHandle[i],      // Thread handle
                        NULL,                  // Default attributes
                        tnsPgm,                // Start routine
                        (void *)&tnsPgmParm);  // Start routine parms
    if (rc != 0)
      printf("pthread_create() failed, rc = %d\n", rc);
  }

  // Wait for each thread to terminate
  for (i=0; i < threads; i++)
  {
    rc=pthread_join(threadHandle[i],  // Thread handle
                    NULL);            // No exit status
  }

}  /* end of Main */
```

Performance data files

Performance data is a set of information about the operation of a system (or network of systems) that can be used to understand response time and throughput. You can use performance data to make adjustments to programs, system attributes, and operations. These adjustments can improve response times and throughputs. Adjustments can also help you to predict the effects of certain changes to the system, operation, or program.

Collection Services collects performance data into a management collection object (*MGTCOL). The Create Performance Data (CRTPFRDTA) command processes data from that collection object and stores the result into performance database files. The database files are divided into the following categories:

Performance data files containing time interval data
These files contain performance data that is collected each interval. See Performance data files containing time interval data for a list of these files, with a brief description and a link to complete information about each file. To understand where the data in these files comes from, refer to system category and file relationships. When viewing these files, you might also find the file abbreviations useful.

Configuration data files
Configuration data is collected once per session. To understand where the data in these files comes from, refer to system category and file relationships. You can find the QAPMCONF, QAPMHDWR, and QAPMSBSD files in the configuration data files.

Trace data files
Trace data is collected only when you choose to do so. You can find the QAPMDMPT file in the trace data files.

Additional field information such as number of bytes and buffer position is available by using the Display File Field Description (DSPFFD) command available on the system. For example, type the following on any command line:

```
DSPFFD file(QSYS/QAPMCONF)
```

For more information about iSeries performance, see Performance.

Performance data files containing time interval data

To view complete information about a performance data file, select the file you want to view from the list below (shown in alphabetical order).

File	Description
QAPMAPPN	APPN data
QAPMASYN	Asynchronous statistics (one per link)
QAPMBSC	Binary synchronous statistics (one per link)
QAPMBUS	Bus counters (one per bus)
QAPMCIOP	Communications IOP data (one per IOP)
QAPMDDI	Distributed Digital Interface (DDI) data (one per link)
QAPMDIOP	Storage device IOP data (one per IOP)
QAPMDISK	Disk storage data (one per read/write head)
QAPMDOMINO	Domino for iSeries data (one record per domino server)
QAPMECL	Token-ring file entries (one per link)
QAPMETH	Ethernet statistics (one per link)
QAPMFRLY	Frame relay data (one per link)
QAPMHDLC	HDLC statistics (one per link)
QAPMHTTPB	Basic data for IBM HTTP server (powered by Apache) (one per server)
QAPMHTTPD	Detailed data for IBM HTTP server (powered by Apache) (one per server component)
QAPMIDLC	Integrated services digital network data link control file entries (one per link)
QAPMIOPD	Extended IOP data (additional data for special IOPs) (one per IOP)
QAPMJOBMI	MI job data (one record per job, task, or thread). You might find information about task type extenders useful when using this document
QAPMJOBOS	Job operating system data (one record per job)
QAPMJOBS and QAPMJOBL	Job data (one record per job, task, or thread)
QAPMJOBWT	Job, task, and thread wait conditions
QAPMJOBWTD	A description of the counter sets found in file QAPMJOBWT.
QAPMJSUM	Job summary data by job group (one record per job group)
QAPMLAPD	Integrated services digital network LAPD file entries (one per link)
QAPMLIOP	Twinaxial workstation controller data (one per physical controller)
QAPMMIOP	Multifunction IOP (one per IOP)
QAPMPOOL and QAPMPOOLL	Main storage data (one per system storage pool)
QAPMPOOLB	Storage pool data (one per pool)
QAPMPOOLT	Storage pool tuning data (one per pool)
QAPMPPP	Point-to-Point Protocol data (one per link)
QAPMRESP	Local workstation response time (one per workstation)
QAPMRWS	Remote workstation response time
QAPMSAP	TRLAN, Ethernet, DDI, and Frame Relay SAP file entries (one per SAP entry)
QAPMSNA	SNA data
QAPMSNADS	SNADS data (one per SNADS job)
QAPMSTND	DDI station data

File	Description
QAPMSTNE	Ethernet station file entries
QAPMSTNL	Token-ring station file entries
QAPMSTNY	Frame relay station file entries
QAPMSYS and QAPMSYSL	System performance data
QAPMSYSCPU	System CPU usage data
QAPMSYSTEM	System-level performance data
QAPMTCP	TCP/IP data
QAPMTCPIFC	TCP/IP data for individual TCP/IP interfaces
QAPMUSRTNS	User-defined transaction data (Each job has one record for each type of transaction)
QAPMX25	X.25 statistics (one per link)

Performance data files: File abbreviations

The performance data files use abbreviations in the field and byte data tables. These abbreviations include:

Abbreviation	Description
Primary files	These files are related to and generated from the category.
C	Character in the Attributes column.
PD	Packed decimal in the Attributes column.
Z	Zoned decimal in the Attributes column.
IOP	Input/output processor or I/O processor. The processors that control the activity between the host system and other devices, such as disks, display stations, and communication lines.
DCE	Data circuit-terminating equipment.
MAC	Medium-access control. An entity in the communications IOP.
LLC	Logical link control. An entity in the communications IOP.
Beacon frame	A frame that is sent when the ring is inoperable.
Type II frame	A connection-oriented frame (information frame) used by Systems Network Architecture (SNA).
I-frame	An information frame.

Performance data files: Collection Services system category and file relationships

When you collect performance data using Collection Services, the data is stored in a management collection (*MGTCOL) object. The CRTPFRDTA command exports data from that management collection object and then writes the data to the performance data files. Each type of data that can be independently controlled and collected by Collection Services is represented by a data category. Each data category contains or provides data that is written to one or more performance data files. For a database file or member to be created, the category (or group of categories) that the file or member is dependent on must exist and be processed by CRTPFRDTA. The table below identifies the category-to-file relationships. There are three types of relationships:

Relationship	Description
Primary files	These files are related to and generated from the category.
Compatibility files	These files are logical files that join primary files to provide performance database compatibility with the previous file structure. If the system generates all participating files (primary categories), compatibility files are also generated.

Relationship	Description
Secondary files	These files are related to and contain some data that is derived from data contained in the category or in the primary file. However, they are not controlled by that category.

Users should note the following:

1. The CRTPFRDTA command generates a database file only when that file is a primary file for the selected category.
2. If a primary file is listed for multiple categories, you must select each of those categories to generate the file.
3. If a primary file for one category is listed as a secondary file for another category, you must select the second category to ensure complete information in your generated database file. For example, as shown in the table below, to generate a complete database file for QAPMECL, you must select both *CMNBASE and *CMNSTN.
4. The system generates compatibility files only when it generates all associated primary files.

The table below illustrates the relationships between system categories and performance database files.

Category	Primary files	Compatibility files	Secondary files
*SYSBUS	QAPMBUS		
*POOL	QAPMPOOLB	QAPMPOOLL	
*POOLTUNE	QAPMPOOLT	QAPMPOOLL	
*HDWCFG	QAPMHDWR		
*SUBSYSTEM	QAPMSBSD		
*SYSCPU	QAPMSYSCPU	QAPMSYSL	
*SYSLVL	QAPMSYSTEM	QAPMSYSL	
*JOBMI	QAPMJOBMI QAPMJOBWT QAPMJOBWTD QAPMJSUM	QAPMJOBL QAPMSYSL	QAPMSYSTEM
*JOBOS	QAPMJOBOS QAPMJSUM	QAPMJOBL QAPMSYSL	QAPMSYSTEM
*SNADS	QAPMSNADS		
*DISK	QAPMDISK		QAPMSYSTEM
*IOPBASE	QAPMLIOP QAPMDIOP QAPMCIOP QAPMMIOP		
*IPCS	QAPMIOPD QAPMTSK		

*CMNBASE	QAPMASYN QAPMBSC QAPMDDI QAPMECL QAPMETH QAPMFRLY QAPMHDLC QAPMIDLC QAPMLAPD QAPMPPP QAPMX25		
*CMNSTN	QAPMSTND QAPMSTNE QAPMSTNL QAPMSTNY none		QAPMDDI QAPMETH QAPMECL QAPMFRLY QAPMX25
*CMNSAP	QAPMSAP		
*LCLRSP	QAPMRESP		
*APPN	QAPMAPPN		
*SNA	QAPMSNA		
*EACACHE	none		QAPMDISK (see note)
*TCPBASE	QAPMTCP		
*TCPIFC	QAPMTCPIFC		
*DOMINO	QAPMDOMINO		
*HTTP	QAPMHTTPB QAPMHTTPD		
*USRTNS	QAPMUSRTNS		
Note: This category is not selectable by CRTPFRDTA. However, it causes additional data to be reported by the *DISK category.			

Performance data files: Field data for configuration database files

Configuration data is collected once per session. The following performance data files show the file names, brief descriptions, and references to field data detail (when provided) for the system configuration data, subsystem data, and hardware configuration data. For information about how Collection Services generates this file and where the data comes from, refer to system category and file relationships.

Field Name	Description
QAPMCONF	System configuration data.
QAPMHDWR	System hardware configuration.
QAPMSBSD	Subsystem data. No field and byte data.

To find more information about this topic, refer to the performance database files overview.

Performance database files: Field data for trace database files

Trace data includes internal system trace data. This is detailed data that you collect to gain additional information about specific jobs and transactions. This type of data should not be collected unless you use the Performance Tools licensed program to analyze it. The following are performance data files that the system supports when using the Start Performance Trace (STRPFRTRC) command.

File Name	Description
QAPMDMPT	System trace data (no field or byte detail).

To find more information about this topic, refer to the performance database files overview.

iSeries Navigator monitors

The monitors included in iSeries Navigator use Collection Services data to track the elements of system performance of specific interest to you. Moreover, they can take specified actions when certain events, such as the percentage of CPU utilization or the status of a job, occur. You can use monitors to see and manage system performance as it happens across multiple systems and groups of systems.

With the monitors, you can start a monitor, and then turn to other tasks on your server, in iSeries Navigator, or on your PC. In fact, you could even turn your PC off. iSeries Navigator continues monitoring and performing any threshold commands or actions you specified. Your monitor runs until you stop it. You can also use monitors to manage performance remotely by accessing them with iSeries Navigator for Wireless.

iSeries Navigator provides the following types of monitors:

System monitor
Collect and display performance data as it happens or up to 1 hour. Detailed graphs help you visualize what is going on with your servers as it happens. Choose from a variety of metrics (performance measurements) to pinpoint specific aspects of system performance. For example, if you are monitoring the average CPU utilization on your server, you can click any collection point on the graph to see a details chart that shows the 20 jobs with the highest CPU utilization. Then, you can right-click any of these jobs to work directly with the job.

Job monitor
Monitor a job or a list of jobs based on job name, job user, job type, subsystem, or server type. Choose from a variety of metrics to monitor the performance, status, or error messages for a job. To work directly with a job, just right-click the job from the list that is shown in the Job Monitor window.

Message monitor
Find out whether your application completes successfully or monitor for specific messages that are critical to your business needs. From the Message Monitor window, you can see the details of a message, reply to a message, send a message, and delete a message.

B2B activity monitor
If you have an application like Connect for iSeries configured, you can use a B2B activity monitor to monitor your B2B transactions. You can view a graph of active transactions over time, and you can run commands automatically when thresholds are triggered. You can search for and display a specific transaction as well as view a bar graph of the detailed steps of that specific transaction.

File monitor
Monitor one or more selected files for a specified text string, for a specified size, or for any modification to the file.

To find out more about monitors, see the following topics:

Monitor concepts
Monitors can display real-time performance data. Additionally, they can continually monitor your system in order to run a selected command when a specified threshold is reached. Learn how monitors work, what they can monitor, and how they can respond to a given performance situation.

Configure a monitor
You can configure a monitor in iSeries Navigator. Use this topic to learn how to set up a monitor and how to configure it to take the best advantage of the available options.

Scenarios
This topic provides scenarios that show how you can use some of the different types of monitors to look at specific aspects of your system's performance.

Monitor concepts

The system monitors display the data stored in the collection objects that are generated and maintained by Collection Services. The system monitors display data as it is collected, for up to one hour. To view longer periods of data, you should use Graph history. You can change the frequency of the data collection in the monitor properties, which overrides the settings in Collection Services.

You can use monitors to track and research many different elements of system performance and can have many different monitors running simultaneously. When used together, the monitors provide a sophisticated tool for observing and managing system performance. For example, when implementing a new interactive application, you might use a system monitor to prioritize a job's resource utilization, a job monitor to watch for and handle any problematic jobs, and a message monitor to alert you if a specified message occurs on any of your systems.

Setting thresholds and actions

When you create a new monitor, you can specify actions you want to occur when the system metric reaches a specified threshold level, or an event occurs. When threshold levels or events occur, you can choose to run an OS/400 command on the endpoint systems, such as sending a message or holding a job queue. Additionally, you may choose to have the monitor carry out several predefined actions such as updating the event log and alerting you by either sounding an alarm on your PC or launching the monitor. Finally, you can automatically reset the monitor by specifying a second threshold level which causes the monitor to resume normal activity when it is reached.

Configure a monitor

System monitors are highly interactive tools that you can use to gather and display real-time performance data from your endpoint systems. Creating a new monitor is a quick and easy process that begins at the **New Monitor** window:

1. In iSeries Navigator, expand Management Central, select **Monitors**, right-click **System**, and then select **New Monitor**.
2. Specify a monitor name. From the **New Monitor-General** page specify a name for your monitor. Provide a brief description so you can find the monitor in a list of monitors.
3. Select metrics. Use the **New Monitor-Metrics** page to select your metrics. You can monitor any number of metrics on any number of endpoint systems or system groups.
4. View and change your metric information. Use the **New Monitor-Metrics** page to edit the properties for each metric. You can edit the collection interval, maximum graphing value, and display time for each metric you select.
5. Set threshold commands. Use the **Thresholds** tab on the **Metrics** page to enable thresholds and specify commands to run on the endpoint system whenever thresholds are triggered or reset.
6. Set threshold actions. Use the **New Monitor-Actions** page to specify the actions you want to occur when a metric threshold is triggered or reset.
7. Select your systems and groups. Use the **New Monitor-Systems and Groups** page to select the endpoint systems or system groups where you want to start a monitor.

After you have created your monitor, right-click the monitor name and select **Start** to run the monitor and begin working with monitor graphs.

Monitor metrics

To effectively monitor system performance, you must decide which aspects of system performance you want to monitor. Management Central offers a variety of performance measurements, known as **metrics**, to help you pinpoint different aspects of system performance.

The **Metrics** page in the **New Monitor** window allows you to view and change the metrics that you want to monitor. To access this page, select **Monitors**, right-click **System**, and then select **New Monitor**. Fill in the required fields, and then click the **Metrics** tab.

When you configure a monitor, you can use any metric, a group of metrics, or all the metrics from the list to be included in your monitor. Metric types you can use in your monitor include the following:

Metric groups:	**Metric description:**
CPU Utilization	The percentage of available processing unit time consumed by jobs on your system. Choose from the following types of CPU Utilization metrics for use in your monitors: • CPU Utilization (Average) • CPU Utilization (Interactive Jobs) • CPU Utilization (Interactive Feature) • CPU Utilization (Database Capability) • CPU Utilization (Secondary Workloads) • CPU Utilization Basic (Average) To learn more about these metrics and how to use them, see the online help available on the **General** tab of the **New Monitor** window or the **Monitor Properties** window in iSeries Navigator.
Interactive Response Time (Average and Maximum)	The response time that interactive jobs experience on your system.
Transaction Rate (Average)	The number of transactions per second completed by all jobs on your system.
Transaction Rate (Interactive)	The number of transactions per second completed on your system by the following types of jobs: • Interactive • Multiple requester terminal (MRT) • System/36 environment interactive • Pass-through
Batch Logical Database I/O	The average number of logical database input/output (I/O) operations currently performed by batch jobs on the system.
Disk Arm Utilization (Average and Maximum)	The percentage of disk arm capacity currently used on your system during the time you collect the data.
Disk Storage (Average and Maximum)	The percentage of disk arm storage that is full on your system during the time you collect the data.
Disk IOP Utilization (Average and Maximum)	How busy the disk input/output processors (IOPs) are on your system during the time you collect the data.
Communications IOP Utilization (Maximum and Average)	How busy the communications input/output processors (IOPs) are on your system during the time you collect the data.

Communications Line Utilization (Average and Maximum)	The amount of data that was actually sent and received on all your system communication lines.
LAN Utilization (Maximum and Average)	The amount of data that was actually sent and received on all your local area network (LAN) communication lines.
Machine Pool Faults	The number of faults per second occurring in the machine pool on the system.
User Pool Faults (Maximum and Average)	The number of faults per second occurring in all of the user pools on the system.

If you need more help, click the **Help** button on the **New Monitor-Metrics** window. Once you become familiar with the Management Central metrics, which metrics you select will depend on the information needs of your computing environment. After you have selected metrics that target the information you are trying to see, you are ready to view and change detailed metric information for each metric you selected for your monitor.

Scenarios: iSeries Navigator monitors

The monitors included in iSeries Navigator provide a powerful set of tools for researching and managing system performance. For an overview of the types of monitors provided by iSeries Navigator, see iSeries Navigator monitors.

For detailed usage examples and sample configurations, see the following scenarios:

Scenario: System monitor
See an example system monitor that alerts you if the CPU utilization gets too high and temporarily holds any lower priority jobs until more resources become available.

Scenario: Message monitor
See an example message monitor that displays any inquiry messages for your message queue that occur on any of your iSeries servers. The monitor opens and displays the message as soon as it is detected.

Scenario: Job monitor
See an example job monitor that tracks the CPU utilization of a specified job and alerts the job's owner if CPU utilization gets too high.

Scenario: System monitor:

Situation

As a system administrator, you need to ensure that the iSeries system has enough resources to meet the current demands of your users and business requirements. For your system, CPU utilization is a particularly important concern. You would like the system to alert you if the CPU utilization gets too high and to temporarily hold any lower priority jobs until more resources become available.

To accomplish this, you can set up a system monitor that sends you a message if CPU utilization exceeds 80%. Moreover, it can also hold all the jobs in the QBATCH job queue until CPU utilization drops to 60%, at which point the jobs are released, and normal operations resume.

Configuration example

To set up a system monitor, you need to define what metrics you want to track and what you want the monitor to do when the metrics reach specified levels. To define a system monitor that accomplishes this goal, complete the following steps:

1. In iSeries Navigator, expand **Management Central** > **Monitors**, right-click **System Monitor**, and select **New Monitor...**
2. On the **General** page, enter a name and description for this monitor.
3. Click the **Metrics** tab, and enter the following values:
   a. Select the **CPU Utilization Basic (Average)**, from the list of Available Metrics, and click **Add**. CPU Utilization Basic (Average) is now listed under Metrics to monitor, and the bottom portion of the window displays the settings for this metric.
   b. For **Collection interval**, specify how often you would like to collect this data. This will override the Collection Services setting. For this example, specify **30 seconds.**
   c. To change the scale for the vertical axis of the monitor's graph for this metric, change the **Maximum graphing value**. To change the scale for the horizontal axis of the graph for this metric, change the value for **Display time**.
   d. Click the **Threshold 1** tab for the metrics settings, and enter the following values to send an inquiry message if the CPU Utilization is greater than or equal to 80%:
      1) Select **Enable threshold.**
      2) For the threshold trigger value, specify **>= 80** (greater than or equal to 80 percent busy).
      3) For **Duration**, specify **1** interval.
      4) For the **OS/400 command**, specify the following:
         ```
         SNDMSG MSG('Warning,CPU...') TOUSR(*SYSOPR) MSGTYPE(*INQ)
         ```
      5) For the threshold reset value, specify **< 60** (less than 60 percent busy). This will reset the monitor when CPU utilization falls below 60%.
   e. Click the **Threshold 2** tab, and enter the following values to hold all the jobs in the QBATCH job queue when CPU utilization stays above 80% for five collection intervals:
      1) Select **Enable threshold.**
      2) For the threshold trigger value, specify **>= 80** (greater than or equal to 80 percent busy).
      3) For **Duration**, specify **5** intervals.
      4) For the **OS/400 command**, specify the following:
         ```
         HLDJOBQ JOBQ(QBATCH)
         ```
      5) For the threshold reset value, specify **< 60** (less than 60 percent busy). This will reset the monitor when CPU utilization falls below 60%.
      6) For **Duration**, specify **5** intervals.
      7) For the **OS/400 command**, specify the following:
         ```
         RLSJOBQ JOBQ(QBATCH)
         ```

         This command releases the QBATCH job queue when CPU utilization stays below 60% for 5 collection intervals.
4. Click the **Actions** tab, and select **Log event** in both the **Trigger** and **Reset** columns. This action creates an entry in the event log when the thresholds are triggered and reset.
5. Click the **Systems and groups** tab to specify the systems and groups you want to monitor.
6. Click **OK** to save the monitor.
7. From the list of system monitors, right-click the new monitor and select **Start.**

Results

The new monitor displays the CPU utilization, with new data points being added every 30 seconds, according to the specified collection interval. The monitor automatically carries out the specified threshold actions, even if your PC is turned off, whenever CPU utilization reaches 80%.

Note: This monitor tracks only CPU utilization. However, you can include any number of the available metrics in the same monitor, and each metric can have its own threshold values and actions. You can also have several system monitors that run at the same time.

Scenario: Job monitor:

Situation

You are currently running a new application on your iSeries server, and you are concerned that some of the new interactive jobs are consuming an unacceptable amount of resources. You would like the owners of the offending jobs to be notified if their jobs ever consume too much of the CPU capacity.

You can set up a job monitor to watch for the jobs from the new application and send a message if a job consumes more than 30% of the CPU capacity.

Configuration example

To set up a job monitor, you need to define which jobs to watch for, what job attributes to watch for, and what the monitor should do when the specified job attributes are detected. To set up a job monitor that accomplishes this goal, complete the following steps:

1. In iSeries Navigator, expand **Management Central > Monitors**, right-click **Job monitor**, and select **New Monitor...**
2. On the **General** page, enter the following values:
   a. Specify a name and description for this monitor.
   b. On the **Jobs to monitor** tab, enter the following values:
      1) For the **Job name**, specify the name of the job you want to watch for (for example, MKWIDGET).
      2) For **Subsystem**, specify QINTER.
      3) Click **Add.**
3. Click the **Metrics** tab, and enter the following information:
   a. In the **Available metrics** list, expand **Summary Numeric Values**, select **CPU Percent Utilization**, and click **Add.**
   b. On the **Threshold 1** tab for the metrics settings, enter the following values:
      1) Select **Enable trigger.**
      2) For the threshold trigger value, specify **>= 30** (greater than or equal to 30 percent busy).
      3) For **Duration**, specify **1** interval.
      4) For the **OS/400 trigger command**, specify the following:
         ```
         SNDMSG MSG('Your job is exceeding 30% CPU capacity') TOUSR(&OWNER)
         ```
      5) Click **Enable reset**.
      6) For the threshold reset value, specify **< 20** (less than 20 percent busy).
4. Click the **Collection Interval** tab, and select **15 seconds.** This will override the Collection Services setting.
5. Click the **Actions** tab, and select **Log event** in both the **Trigger** and **Reset** columns.
6. Click the **Servers and groups** tab, and select the servers and groups you want to monitor for this job.
7. Click **OK** to save the new monitor.

8. From the list of job monitors, right-click the new monitor and select **Start.**

Results

The new monitor checks the QINTER subsystem every 15 seconds, and if the job MKWIDGET is consuming more than 30 percent of the CPU, the monitor sends a message to the job's owner. The monitor resets when the job uses less than 20% CPU capacity.

Scenario: Message monitor:

Situation

You company has several iSeries servers running, and it is time-consuming to check your message queue for each system. As a system administrator, you need to be aware of inquiry messages as they occur across your system.

You can set up a message monitor to display any inquiry messages for your message queue that occur on any of your iSeries systems. The monitor opens and displays the message as soon as it is detected.

Configuration example

To set up a message monitor, you need to define the types of messages you would like to watch for and what you would like the monitor to do when these messages occur. To set up a message monitor that accomplishes this goal, complete the following steps:

1. In iSeries Navigator, expand **Management Central > Monitors**, right-click **Message monitor**, and select **New Monitor...**
2. On the **General** page, enter a name and description for this monitor.
3. Click the **Messages** tab, and enter the following values:
   a. For **Message queue to monitor**, specify **QSYSOPR.**
   b. On the **Message set 1** tab, select **Inquiry** for **Type**, and click **Add**.
   c. Select **Trigger at the following message count**, and specify **1** message.
4. Click the **Collection Interval** tab, and select **15 seconds.**
5. Click the **Actions** tab, and select **Open monitor.**
6. Click the **Systems and groups** tab, and select the systems and groups you would like to monitor for inquiry messages.
7. Click **OK** to save the new monitor.
8. From the list of message monitors, right-click the new monitor and select **Start.**

Results

The new message monitor displays any inquiry messages sent to QSYSOPR on any of the iSeries servers that are monitored.

Note: This monitor responds to only inquiry messages sent to QSYSOPR. However, you can include two different sets of messages in a single monitor, and you can have several message monitors that run at the same time. Message monitors can also carry out OS/400 commands when specified messages are received.

Graph history

Graph history provides a graphical view of performance data collected over days, weeks, months, or years with Collection Services. You do not need to have a system monitor running to view performance data. As long as you use Collection Services to collect data, you can view the Graph History window.

- **Graph history concepts**
  The amount of historical data available to graph history depends largely on the collection retention value in Collection Services, and on whether PM/400 is enabled. See this topic for a description of the available options for managing and displaying records of performance data.
- **Use graph history**
  Graph history is accessible through iSeries Navigator. See this topic for step-by-step instructions.

For more information about monitoring system performance, see the Track performance data topic.

Graph history concepts

Graph history displays data contained in the collection objects created by Collection Services. Therefore, the type and amount of data available is dependent on your Collection Services configuration.

The amount of data that is available to be graphed is determined by the settings that you selected from the Collection Services properties, specifically the collection retention period. Use iSeries Navigator to activate PM/400 over multiple systems. When you activate PM/400, you can use the graph history function to see data that was collected days ago, weeks ago, or months ago. You go beyond the real-time monitor capabilities, and have access to summary or detailed data. Without PM/400 enabled, the graph data field supports 1 to 7 days. With PM/400 enabled, you define how long your management collection objects remain on the system:

- **Detailed data**
  The length of time that management collection objects remain in the file system before they are deleted. You can select a specific time period in hours or days, or you can select **Permanent**. If you select **Permanent**, the management collection objects will not be automatically deleted.
- **Graph data**
  The length of time that the details and properties data that is shown in the Graph History window remains in the system before it is deleted. If you do not start PM/400, you can specify one to seven days. If you do start PM/400, you can specify 1 to 30 days. The default is one hour.
- **Summary data**
  The length of time that the data collection points of a graph can be displayed in the Graph History window or remain in the system before they are deleted. No details or properties data is available. You must start PM/400 to enable the summary data fields. The default is one month.

Use graph history

Graph history is included in iSeries Navigator. To view the graph history of the data that you are monitoring with Collection Services, do these steps:

1. Follow the iSeries Navigator online help for starting Collection Services on either a single system or on a system group.
2. From the **Start Collection Services - General** page, select **Start Performance Management/400** if needed.
3. Make changes to the other values for the collection retention period.
4. Click **OK**.
5. You can view the graph history by right-clicking either a system monitor or a Collection Services object and selecting **Graph History**.
6. Click **Refresh** to see the graphical view.

Once you have launched a graph history, a window displays a series of graphed collection points. These collection points on the graph line are identified by three different graphics that correspond to the three levels of data that are available:

- A square collection point represents data that includes both the detailed information and properties information.
- A triangular collection point represents summarized data that contains detailed information.
- A circular collection point represents data that contains no detailed information or properties information.

Performance Management/400

Performance Management/400 (PM/400) is automated and self-managing, which allows for easy use. PM/400 automatically triggers Collection Services to gather the nonproprietary performance and capacity data from your server and then sends the data to IBM. All collection sites are network secure, and the time of the transfer is completely under your control. When you send your data to IBM, you eliminate the need to store all the trending data yourself. IBM stores the data for you and provides you with a series of reports and graphs that show your server's growth and performance. You can access your reports electronically using a traditional browser. These reports can help you plan for and manage system resources through ongoing analysis of key performance indicators.

The IBM Operational Support Services for PM/400e offering includes a set of reports, graphs, and profiles that help you maximize current application and hardware performance (by using performance trend analysis). This offering also helps you better understand (by using capacity planning) how your business trends relate to the timing of required hardware upgrades such as CPU or disk. Capacity planning information is provided by trending system utilization resources and throughput data, which can be thought of as an early warning system for your servers. Think of PM/400e as a virtual resource that informs you about the "health" of your system.

PM/400 uses less than 1 percent of your central processing unit (CPU). It uses approximately 58 MB of disk space, which depends on your hardware model and the size of your collection intervals.

PM/400 concepts
Learn about the functions and benefits PM/400 can provide and about important implementation considerations.

Configure PM/400
To start using PM/400, you need to activate it, set up a transmission method that allows you to send data and receive reports, and, finally, customize data collection and storage.

Manage PM/400
Now that you have set up your network, you can perform a variety of tasks with PM/400.

PM/400 reports
The iSeries server can be configured to send Collection Services data directly to IBM with PM/400. IBM then generates several reports that you can either view on the Web or have sent directly to you. Activating PM/400 to automatically generate your reports not only saves you time and resources, but also allows you to plan ahead by forecasting your future growth needs.

PM/400 concepts

PM/400 uses Collection Services to gather the nonproprietary performance and capacity data from your server and then sends the data to IBM. This information can include CPU utilization and disk capacity, response time, throughput, application and user usage. When you send your data to IBM, you eliminate the need to store all the trending data yourself. IBM stores the data for you and provides you with a series of reports and graphs that show your server's growth and performance. You can access your reports electronically using a traditional browser.

Benefits of PM/400
PM/400 can help make managing system resources and capacity planning significantly easier. Learn more specific ways to utilize PM/400.

PM/400 offering options
PM/400 offers a wide range of options. Use this information to decide which combination of services best suits your needs.

Data collection considerations
PM/400 uses Collection Services to gather performance data. Learn how PM/400 and Collection Services work together to provide the data you need.

Benefits of PM/400: When you use the PM/400 service, you receive these benefits:

- **Helps you avoid unfortunate surprises.**
  You avoid disappointing surprises. You are in control over managing the growth and performance of your system, which means that you manage the system. Your system does not manage you.
- **Saves you time.**
  You eliminate the labor intensive and expensive tasks of collecting and reporting performance data by doing it automatically. This benefit gives you the advantage of focusing your resources on managing your system and applications.
- **Allows you to plan ahead for maximum efficiency.**
  You can proactively plan for the financial requirements to keep your system running at its peak efficiency.
- **Provides easy to understand information.**
  You understand the information, which in turn makes it easy for you to present to senior management when asked the question, "Why do we need to upgrade?"
- **Allows you to forecast the future.**
  You can make projections of your data processing growth based on factual trend information.
- **Allows you to identify system problems.**
  PM/400 data enables you to identify performance bottlenecks.
- **Allows you to help estimate the size of your next upgrade.**
  You can upload PM/400 data to the Workload Estimator for iSeries

  for sizing your next upgrade.

See Performance Management/400 to learn more about what you need to do before using PM/400.

Operational Support Services for PM/400e offering: You can receive your graphs and reports either electronically or in printed form. You can receive electronic graphs monthly. You receive printed graphs monthly or quarterly. The PM/400e service fee varies depending on how often you choose to receive your performance information and your choice of format, electronically or in printed form. Some of these report options are free, and some are not. The marketing and services organizations in each country can give you details on the support that is available. Visit the PM/400e Web site

for information on the free and fee options.

See Performance Management/400 to learn more about what you need to do before using PM/400.

Data collection considerations for PM/400: The most important requirement for establishing an accurate trend of the system utilization, workload, and performance measurements is consistency. Ideally, performance data should be collected 24 hours per day. Because of the relationship between PM/400 and Collection Services, you need to be aware of the implications that can occur when you are using PM/400.

Here are some guidelines to help you define your collections when you are using PM/400:

- **Select the QMPGDATA library to store your data.**
  The **Location to store collections** field uses the default value /QSYS.LIB/QMPGDATA.LIB when PM/400 is active. If you replace QMPGDATA with some other value, PM/400 cycles the collection on the hour and changes it back to QMPGDATA. If you want to collect data into a different library, change where PM/400 looks for data. Type **GO PM400** from the command line, select option 3 (Work with Customization), and change the library name.
- **Collect data continuously with Collection Services.**
  PM/400 satisfies this requirement by collecting data 24 hours a day with Collection Services. PM/400 collects performance data at 15-minute intervals. PM/400 uses the 15-minute interval default, but does not change what the interval is set to. A 15-minute interval is the recommended interval.
- **Select the Standard plus protocol profile.**
  Standard plus protocol is the default value for the collection profile. The collection profile indicates what data is collected. The data categories in the standard plus protocol profile correspond to the *ALL value for the DATA parameter on the Start Performance Monitor (STRPFRMON) command. If you change this to any other value, PM/400 changes it back on the hour. This is true even if you select Custom and include all categories. The change takes effect immediately. The collection does not cycle (unless required to do so for other reasons). This action is done to gather enough information for PM/400 reports.
- **Avoid making interim changes to collection parameters when PM/400 is active.**
  For example, when you activate PM/400, the **Create database files during collection** field is checked as the default value. If you change this, PM/400 changes it back to the default value on the hour. The change takes effect immediately. The collection does not cycle (unless required to do so for other reasons).
- **Ending Collection Services**
  You can end Collection Services at any time from iSeries Navigator. If you end Collection Services, the following considerations apply when PM/400 is running:
  - The PM/400 scheduler starts Collection Services at the beginning of the next hour.
  - Days with little or no data collected are not included in trend calculations. Therefore, you should not interrupt Collection Services often.

If you do not want to start Collection Services, you can temporarily turn off PM/400.

Configure PM/400

PM/400 automates the collection of performance data through Collection Services. You can specify which library to put the data in as long as the library resides on the base auxiliary storage pool (ASP). The library should not be moved to an independent auxiliary storage pool because an independent auxiliary storage pool can be varied off, which stops the PM/400 collection process. PM/400 creates the library during activation if the library does not already exist.

To begin using PM/400, you need to perform the following tasks:

Activate PM/400
PM/400 ships with OS/400, but you must activate it to use its collecting capabilities.

Determine which transmission method to use
Determine how you want to send data. You can either gather your data with Management Central and send the data with Electronic Service Agent (Extreme Support) or you can have PM/400 gather the data and send it over the SNA protocol.

Customize PM/400
Now that you have set up your network, you may need to customize PM/400 to fit your needs.

Activate Performance Management/400: You must start PM/400 to take advantage of its data collecting capabilities. You can start PM/400 by using any one of the following methods:

Use iSeries Navigator
Use iSeries Navigator to activate PM/400 over multiple systems. When you activate PM/400, you can use the Graph History function to see data that was collected days ago, weeks ago, or months ago. You go beyond the real-time monitor capabilities. You have access to summary data or detailed data. Without PM/400 enabled, the graph data field supports 1 to 7 days. With PM/400 enabled, you choose the length of time to retain the data.

To start PM/400 from iSeries Navigator, do the following steps:

1. In iSeries Navigator, expand the system where you want to start PM/400.
2. Expand **Configuration and Service**.
3. Right-click **Collection Services**.
4. Select **Performance Management/400**.
5. Select **Start**.
6. Select the systems on which you want to start PM/400.
7. Click **OK**.

Reply to message CPAB02A in the QSYSOPR message queue
When the QSYSWRK subsystem starts, this message asks if you want to activate PM/400.

1. From the character-based interface, reply with a G to the message in QSYSOPR "Do you want to activate PM/400? (I G C)." QSYSOPR message queue receives the message that PM/400 is activated.
2. Update your contact information. Issue the **GO PM400** command and specify option 1.

Issue the Configure PM/400 (CFGPM400) command
From the character-based interface, you can issue the Configure PM/400 (CFGPM400) command.

You can proceed to the next step in the setup process, which is to determine which transmission method to use to send data to IBM.

For an overview of iSeries performance topics, see Performance.

Determine which PM/400 transmission method to use: Since V5R1, the PM/400 transmission process has taken advantage of the network configuration that you perform with Management Central to set up a central system and endpoint systems. However, you can still use the character-based interface to configure PM/400. Choose which transmission method you want to use:

- Send data with the Electronic Service Agent over Extreme Support
  If you choose this transmission method, you need to configure PM/400 to have data gathered by Management Central. Perform this configuration for PM/400 if your servers have V4R5 or later of the operating system installed (you must also have the Universal Connection fixes applied). You would choose this method if you use Extreme Support.
- Send data with SNA protocol
  If you choose this transmission method, you need to configure PM/400 by using the character-based interface. PM/400 gathers your data and transmits it by using SNA. Perform this configuration for PM/400 if your servers have OS/400 V4R5 or earlier installed.

Once you have implemented which transmission method you want to use, you are ready to do the other tasks to Manage PM/400.

Send PM/400 data with Service Agent over Extreme Support (Universal Connection): PM/400 uses Collection Services to gather the nonproprietary performance and capacity data from your server. After you have collected this data, you can use Service Agent over Extreme Support to send the data to IBM.

To take advantage of these capabilities, you must have V5R1 or V5R2 installed on your servers or V4R5 with the Universal Connection fixes applied. Here are the steps to follow to configure for PM/400:

1. Activate PM/400
   You must start PM/400 to take advantage of its data collecting capabilities.
2. Set up your Management Central network.
   Define which server is your central system and which servers are your endpoint systems. You can use this network hierarchy to send your data from your endpoint systems to a central location before sending the data to IBM.
3. Connect to IBM to transmit your data with the Universal Connection.
   This is the connection that Management Central will use to transmit the PM/400 data to IBM. In previous releases, you used the electronic customer support (ECS) connection that ran over SNA. When you use the Universal Connection, you can transmit your data over TCP/IP.
4. Gather PM/400 performance data.
   Use the Management Central inventory function to gather your data.
5. Send your data to IBM.
   Use the Electronic Service Agent, which is available under Extreme Support in the Management Central hierarchy, to send your data to IBM. The Electronic Service Agent uses the Universal Connection.

You can also send data with the SNA protocol.

Once you have configured PM/400, you are ready to do the other tasks to manage PM/400.

Gather PM/400 performance data: You can use Management Central to gather your PM/400 performance data if you have done the following tasks:
1. Activated PM/400
2. Configured the Universal Connection
3. Set up your Management Central network
4. Verified that the Electronic Service Agent is installed on your system.

To gather PM/400 performance data on an endpoint system or system group, follow these steps:
1. In iSeries Navigator, expand **Management Central**.
2. Expand **Endpoint Systems** or **System Groups**.
3. Right-click an endpoint system or a system group, and select **Inventory**.
4. Select **Collect**.
5. Select one or more inventories to collect. In this case, you would select **PM/400 performance data**.
6. If you want an action to run on the central system when collection completes, select the action from the list.
7. Click **OK** to start collecting the data immediately or click **Schedule** to specify when to collect the data.

Once you have configured your servers, you are ready to do the other tasks to manage PM/400.

Sending data with SNA protocol: If you choose not to take advantage of sending data with the Electronic Service Agent over Extreme Support, you can still use the character-based interface to transmit data. PM/400 asks you a series of questions about the configuration and use of your servers. The Configure PM/400 display asks you questions about how you want your servers to send and receive PM/400 performance data. The first part of the process involves setting up your network. The second part asks you how you want to transmit your data. When you use the character-based interface, you can use a direct dial line to transmit data.

Follow these steps to send data with SNA:
1. Activate PM/400
   You must start PM/400 to take advantage of its data collecting capabilities.
2. Select which network configuration you want to use.

Determine which network configuration you will use to transmit data. Choose how you connect to IBM by using a direct dial line, an existing Internet Service Provider (ISP), or a virtual private network (VPN). If you want to use ISP or VPN, you must configure a Universal Connection.

If you decide to use the direct dial line to report data to IBM, you have several choices as to how you configure your network. Select which configuration is appropriate for your network, and perform the steps outlined for that particular configuration from the Configure PM/400 display:

- As a single server that sends its data directly to IBM.
- As a host server, which means that you want your server to receive performance data from other servers (remote servers) and then forward the data to IBM. The host server cannot be at a release level that is earlier than other servers. In other words, the host server must be at the same release level or later than other servers.
- As a remote server, which means that you can send performance data to a host server. You identify on the Configure PM/400 display that you need a remote server, and then use option 5 (Work with remote iSeries systems) from the PM/400 menu to define your remote servers.

3. Work with remote servers.
   If you choose to set up your network for a host server, you need to identify those servers that will send their data to your host server. You can ignore this step if you are using a single server or a remote server.
4. Customize PM/400.
   After you have configured your network, you need to establish the global parameters for the operation of the PM/400 software. You need to define the PM/400 data telephone number if you would like to connect to IBM with a direct dial line.

Once you have configured your servers, you are ready to do the other tasks for managing PM/400.

PM/400 network for a single server: A single server sends its data directly to IBM. Here are the steps that you need to follow to configure PM/400 for a single server only if PM/400 gathers data and transmits data over SNA. From the Configure PM/400 (CFGPM400) display on your server:

1. Type **CFGPM400** from the command line.
2. Specify *YES for the **Send performance data to IBM** field.
3. Specify *NO for the **Receive performance data** field.
4. Accept the default library of QMPGDATA.
5. If you specify *YES for Send performance data to IBM, you see additional information that indicates whether the appropriate communications objects exist. If the objects do not exist, PM/400 creates the communications objects for you for transmission. Respond appropriately to the additional displays.
6. Type your company's contact information on the Work with Contact Information display.

If you decide that the single server setup is not what you want, you can choose another SNA configuration option.

Once you have configured your servers, you are ready to do the other tasks to manage PM/400.

PM/400 network for a host server: A host server receives performance data from other servers and then forwards the data to IBM. Here are the steps that you need to follow to configure PM/400 for a host server only if PM/400 gathers data and transmits data over SNA:

1. From the Configure PM/400 display on your host server
   - Type **CFGPM400** from the command line.
   - Specify *YES for the **Send performance data to IBM** field.
   - Specify *YES for the **Receive performance data** field.
   - Accept the default library of QMPGDATA.
2. From the Work with Remote iSeries Systems display on your host server

- Press F6 (Create) to identify which servers will send their data to your host server.
- Complete the fields and press Enter.

Note: The following situation occurs only if PM/400 gathers data and transmits data over SNA. If you have a network of systems, it is recommended that you use the Universal Connection and Management Central in iSeries Navigator to gather and transmit the data for those systems.

PM/400 automatically schedules the transmission of data from the primary server to IBM the day after data is received from a remote server. If the automatic scheduling does not fit your work management scheme, you can manually schedule the transmission of the data from the primary server.

Here is a tip that you should keep in mind when scheduling the transmission of your data. Throughout the week, evenly schedule the transmission of data to the primary server. This action minimizes the performance impact on the primary server. For example, in a network of twelve servers, you might have three groups of four systems. You can schedule each group to send their data on Monday, Wednesday, and Friday. This evenly distributes the amount of data that is sent to the primary server.

If you decide that the host server setup is not what you want, you can choose another SNA configuration option.

Once you have configured your servers, you are ready to do the other tasks to manage PM/400.

PM/400 network for a remote server: A remote server sends its performance data to a host server. Here are the steps that you need to follow to configure PM/400 for a remote server only if PM/400 gathers data and transmits data over SNA. From the Configure PM/400 display (CFGPM400) on your remote server, do these steps:

1. Type **CFGPM400** from the command line.
2. Specify *NO for the **Send performance data to IBM** field.
3. Specify *NO for the **Receive performance data** field.
4. Accept the default library of QMPGDATA.

Note: If you have a network of systems, it is recommended that you use the inventory function of iSeries Navigator to gather your data and then transmit the data for those systems over the Universal Connection.

If you decide that the remote server setup is not what you want, you can choose another SNA configuration option.

Once you have configured your servers, you are ready to do the other tasks to manage PM/400.

Work with remote servers: In some sites, a host server in a network sends the required performance data to IBM for processing. When you use a host server network, you have the other servers in the network send their performance data to this host server for transmission to IBM. To set up your network to use a host server, you must identify the other remote servers and set the schedule for their data transmission. The Work with Remote iSeries Systems display enables you to define these other servers.

Notes:

1. You do not have to use this display if you are setting up your network as a remote server or as a single server. You perform this task only if PM/400 gathers data and transmits data over SNA.
2. If you have a network of systems, it is recommended that you use the inventory function of iSeries Navigator to gather your data and then transmit the data for those systems over the Universal Connection.

Follow these steps to define your remote servers:

1. Type **GO PM400** from the command line.

2. Type a 5 (Work with remote iSeries systems) from the PM/400 Menu and press Enter. You do not see a remote server displayed initially. You must create a new remote location.
3. Create a new remote location by pressing F6 (Create).
4. Record the values for the following information. Use the Display Network Attributes (DSPNETA) command to display these values from the remote system.
   - Local network ID
   - Default local location

   The Work with Remote iSeries Systems display shows a list of remote servers. This list includes the status for the servers (active or inactive) and the descriptions for each server.
5. Create or change the description for a remote site server by using the PM/400 Remote Site Maintenance display or the Change Remote Site iSeries display. The remote location name must be unique between remote servers.

PM/400 automatically schedules the transmission of data from the primary server to IBM the day after data is received from a remote server. If the automatic scheduling does not fit your work management scheme, you can manually schedule the transmission of the data from the primary server. To manually schedule the transmission of data, see the PM/400 scheduler.

The PM/400 software assumes that you defined the Advanced Peer-to-Peer Networking (APPN) link between the server that receives data (the host server) and the server that sends data (the remote server). If your system has the system value QCRTAUT (Create default public authority) set to *EXCLUDE or *USE, you should see Create a device description for a remote server for information on how to define your controller descriptions. If your network does not meet these assumptions, see Non-APPN network considerations for information about creating device pairs to support the connection to each remote server.

Once you have defined your remote servers, you are ready to customize PM/400 to use a specific line connection.

Work with remote servers in a non-APPN network: The primary server receives PM/400 data from other servers and then sends the data to IBM. The remote server sends PM/400 data to the primary server. The following information assumes that the controllers that are referred to have previously been defined.

You need to create device pairs to support the connection to each remote server only if PM/400 gathers data and transmits data over SNA.

1. Use the Create Device Description (APPC) (CRTDEVAPPC) command. On the remote server, type CRTDEVAPPC. Press F4 to prompt for the parameters, and define the values with the following information:

Remote system

DEVD(Q1PLOC)	Specifies the name of the device description.
RMTLOCNAME(Q1PLOC)	Specifies the name of the remote location.
ONLINE(*YES)	Specifies whether this device is varied online when the system is started or restarted.
LCLLOCNAME(Q1PRMxxx)	Specifies the local location name. Q1PRMxxx matches the RMTLOCNAME of the primary server, where xxx is unique for each remote location.
CTL(yyyyyy)	Specifies the name of the attached controller, where yyyyyy is a controller that attaches to the primary server.

MODE(Q1PMOD)	Specifies the mode name.
APPN(*NO)	Specifies if the device is APPN-capable.

2. Specify the following information on the primary server. At the command line, type CRTDEVAPPC. Press F4 to prompt for the parameters, and define the values with the following information:

Primary server

DEVD(Q1PRMxxx)	Specifies the name of the device description. The name that is used here matches the device description name for the remote system.
RMTLOCNAME(Q1PRMxxx)	Specifies the name of the remote location. The name that is used here matches the LCLLOCNAME value of the remote server, where xxx is unique for each remote location.
ONLINE(*YES)	Specifies whether this device is varied online when the system is started or restarted.
LCLLOCNAME(Q1PLOC)	Specifies the local location name. This value matches the RMTLOCNAME of the remote server.
CTL(aaaaaa)	Specifies the name of the attached controller, where aaaaaa is a controller that attaches to the remote server.
MODE(Q1PMOD)	Specifies the mode name.
APPN(*NO)	Specifies if device is APPN-capable.

3. Vary on the devices (Vary Configuration (VRYCFG) command) after you define the APPC devices. On the remote server, type VRYCFG. Press F4 to prompt for the parameters.

Vary on remote system

CFGOBJ(Q1PLOC)	Specifies the configuration object.
CFGTYPE(*DEV)	Specifies the type of configuration object.
STATUS(*ON)	Specifies the status

4. Type option 5 on the PM/400 Menu to add Q1PRMxxx as a remote server. See Work with remote servers for instructions on how to add a remote server.

Now that you have finished configuring PM/400, see Manage PM/400 for other tasks that you can perform with PM/400.

Create a device description for PM/400: The following steps are necessary on each remote server that has the Create default public authority (QCRTAUT) system value set to *EXCLUDE or *USE. If QUSER does not have *CHANGE authority to device description Q1PLOC, remote transmissions will fail. These steps ensure that the device will not be created or deleted automatically.

Note: This task is necessary only if PM/400 gathers data and transmits data over SNA.

If you allow the device to be created automatically, the device description is created with PUBLIC *EXCLUDE or *USE authority, depending on the value set for QCRTAUT. Whether a device can be created or deleted automatically is controlled by the controller. Refer to the following commands to determine how these parameters are defined on the system:

- Create Controller Description (APPC) (CRTCTLAPPC) command
- Change Controller Description (APPC) (CHGCTLAPPC) command
- Display Controller Description (DSPCTLD) command

For systems that are not configured to use APPN, see Work with remote servers in a non-APPN environment for information on how to create the device description.

The following information assumes that the controller that will be used to communicate with the host server was defined previously on the remote server.

On the *remote server*, re-create device description Q1PLOC:

```
    VRYCFG  CFGOBJ(Q1PLOC)
            CFGTYPE(*DEV)
            STATUS(*OFF)


    DLTDEVD   DEVD(Q1PLOC)


    CRTDEVAPPC DEVD(Q1PLOC)
              RMTLOCNAME(Q1PLOC)
              ONLINE(*NO)
              LCLLOCNAME(name of remote system)
              RMTNETID(remote netid of primary (or central) system)
              CTL(name of controller that the device will be attached to)
              AUT(*EXCLUDE)



   CRTOBJAUT  OBJ(Q1PLOC)
              OBJTYPE(*DEVD)
              USER(QUSER)
              AUT(*CHANGE)


    VRYCFG     CFGOBJ(Q1PLOC)
               CFGTYPE(*DEV)
               STATUS(*ON)
```

Now that you have finished configuring PM/400, see Manage PM/400 for other tasks that you can perform with PM/400.

Customize PM/400: The Work with PM/400 Customization display provides you with the ability to:

Establish global parameters for the operation of PM/400 software
The global parameters allow you to customize the following items. See the online help for a description of these fields:

- Priority limits
- Trend and shift schedules
- Performance data library
- Removal specifications

Define your PM/400 data telephone number
Outside the United States and Canada, you must provide PM/400 with the telephone number of the

IBM location that will receive your data. For most locations, PM/400 tries to select the correct telephone data number for your location when you initiate the configure PM/400 process.

Vary a line off and on with PM/400
The PM/400 Line Control display allows PM/400 to vary the line off, transmit the PM/400 data, and then put the line back in the connect pending state.

To customize the global parameters, do the following steps:

1. Type **GO PM400** from the command line.
2. Type a 3 from the PM/400 menu to display the Work with PM/400 Customization display and press Enter.

If you are using Collection Services to gather your PM/400 data, you should take into account some data collection considerations for PM/400.

See manage PM/400 for other tasks that you can perform with PM/400.

Verify the PM/400 data number: If your server is using a direct dial connection to IBM, you must verify that the PM/400 telephone number is correct. The telephone number must also contain the correct prefixes for your line.

Note: This is for SNA transmissions only.

To check the format of the telephone number of the electronic customer support line, do the following steps:

1. Type
   ```
   DSPDTAARA DTAARA(QUSRSYS/QESTELE)
   ```
   and press Enter.
2. Determine the connection number prefix found in offset 0. For example, if offset 0 is **’T9:1800xxxxxxx’** the prefix is **T9:**.
3. Type
   ```
   DSPDTAARA DTAARA(QUSRSYS/Q1PGTELE)
   ```
   and press Enter.
4. Offset 0 (zero) is the dialing string that will be used. (The other numbers will not be used.)
5. If you use your ECS line to order PTFs, you can compare the format in offset 0 (zero) to the format used for the ECS line, CALL QESPHONE, make a note of the string being used, and compare it to the value found in step 2.

   The telephone numbers will be different but the prefix should be the same (that is, SST9:1800..., SST:1800...and so forth).

If you need to change your telephone number, use the Change Data Area (CHGDTAARA) command:

Type **CHGDTAARA**, where DTAARA is Q1PGTELE, LIB is QUSRSYS, the substring starting position is *ALL, and the New value is ’SST:18005475497’

Note: The new value should be your dialing prefix, followed by 18005475497 for U.S.A and Canada.

Now that you have completed your PM/400 configuration, see manage PM/400 for the tasks that you can perform.

Set a direct dial line for PM/400: For most locations, PM/400 tries to select the correct data telephone number for your location. You should always verify that the PM/400 data telephone number is correct. If you do not have information that contains the PM/400 data telephone number and the PM/400 support number, contact your local IBM support personnel. They can provide you with the proper telephone numbers.

Note: This telephone number is not required if you are transmitting data through the Universal Connection. You need this number only if you are using the direct dial line.

To define the PM/400 data telephone number or to change the number, do the following steps:

1. Type **GO PM400** from the command line.
2. Type a 3 from the PM/400 Menu to display the Work with PM/400 Customization display and press Enter.
3. On this display, scroll forward until you see the section of the display that shows you the telephone number fields.
4. Type the correct dialing sequence in the **IBM PM/400 phone number** field. For many IBM modems, you are required to use the colon (:) character for dial tone.

Vary a line off and on with PM/400: Sometimes the line that PM/400 uses is in the connect pending state. This state does not allow PM/400 to access the line to transmit data. The PM/400 Line Control display allows PM/400 to vary off the line, transmit the data, and then put the line back in the connect pending state. When you use this display, you can change the PM/400 transmission task (Q1PCM1) to check for line status and vary off the appropriate line. Once the transmission is complete, the same line is placed in a connect pending state.

Note: This task is necessary only if PM/400 gathers data and transmits data over SNA.

To vary off and on a line, do the following steps:

1. Start the PM/400 line monitoring function by typing **PMLINMON** from the command line. You should see the PM/400 Line Control display.
2. Read the warning that is shown on the first display and press Enter.
3. Define the line, controller, and device combinations that PM/400 needs to vary off.
4. Use the prompt **Do you want PM/400 automatic line control active?** as a master control switch for the function. If you specify **YES**, the PM/400 function is active. If you specify **NO**, the function is disabled.

   If you specify **NO**, you do not need to define the line control list again when you specify **YES**. You can vary off and on a line by specifying the line only. You can vary off and on a line, controller, and device by specifying all three descriptions.
5. Verify the line, controller, and device that you defined. Press Enter to see a summary of your choices.
6. Press Enter to confirm your choices or press F12 to return to the previous display to change your entries.

You can also set up PM/400 line control by using the Configure PM/400 (CFGPM400) command.

See manage PM/400 for additional tasks that you can perform with PM/400.

Manage PM/400

After you have set up your network to use PM/400, you can perform the following tasks:

Deactivate PM/400
Learn how you can stop PM/400.

Change contact information
Learn how to change your contact information from the original settings.

Schedule jobs with PM/400
Learn how to schedule jobs with PM/400.

Omit items from PM/400 analysis
Learn how to omit jobs, users, and communications lines when performing an analysis with PM/400.

Turn off PM/400 momentarily
Learn how you can stop PM/400 momentarily.

Display PM/400 status
Learn how to use iSeries Navigator or the PM/400 menu to display PM/400 status.

View PM/400 reports
See examples of the PM/400 reports and explanations of how to interpret those reports.

View graph history of data
Graph history provides a graphical view of performance data that was collected over a specified period of time. Find out how to view this data.

Deactivate PM/400: To stop PM/400 from running, you can use either of the following methods:

With iSeries Navigator

Perform the following steps:
1. In iSeries Navigator, expand the system where PM/400 is running.
2. Expand **Configuration and Service**.
3. Right-click **Collection Services**.
4. Select **Performance Management/400**.
5. Select **Stop**.
6. Select the systems on which you want to stop PM/400.
7. Click **OK**.

With an API

Use the End PM/400 (Q1PENDPM) API to deactivate PM/400.

See Manage PM/400 to learn about other tasks that you can perform.

Change PM/400 contact information: During the configuration of the PM/400 software, you identified the contact person and provided mailing information for your organization. If at a later time, you need to update the information, use the Work with Contact Information option to change that information. To change the contact information, do the following steps:
1. Type **GO PM400** from the command line.
2. Type a 1 from the PM/400 Menu and press Enter. The Work with Contact Information display appears.
3. Change the contact information, as appropriate, and press Enter.

See Manage PM/400 for other tasks that you can perform.

Schedule jobs with PM/400: Integral to the PM/400 software is a scheduler that automatically starts the jobs that are necessary to support the PM/400 performance data collection and analysis.

Part of the PM/400 software activation process includes starting a job that is called Q1PSCH. This job, in turn, starts other jobs as shown in the following table:

To access the PM/400 scheduled jobs, do the following:

1. Type **GO PM400** from the command line.
2. Type a 2 from the PM/400 Menu and press Enter. The Work with Automatically Scheduled Jobs display appears.
3. You can change the status for each job from active to inactive. Type a 2 (Change) next to the job that you want to change and press Enter. You are shown the Change Automatically Scheduled Jobs display.

The following table shows you a list of the possible PM/400 jobs.

PM/400 scheduled jobs

Job	**Schedule**	**Function**
Q1PTEST	At activation	Verifies that PM/400 is activated and then goes to inactive status.
Q1PCM1	Weekly	Transmits the reduced performance data to IBM. This job is active only if you are using a direct dial line.
Q1PCM2	Daily	Varies communications offline.
Q1PPMSUB	Hourly	Ensures that Collection Services is collecting data.
Q1PDR	Daily	Performs data reduction and purges performance data.
Q1PPG	Monthly	Purges reduced performance data.
Q1PCM3	As needed	Varies communications offline after direct dial transmission fails to vary lines off.
Q1PCM4	As needed	Accesses the PM/400 data from remote servers. This job is started only if you have added remote systems by using option 5 from the PM/400 Menu.
Q1PPMCHK	Every 4 hours	Verifies that data collection is active.
Q1PMONTH	Monthly	Allows for monthly transmission if you require an additional transmission during the month. The default value is set to inactive. This job is available only if you are using a direct dial line.

See Manage PM/400 to learn about other tasks that you can perform.

Omit items from PM/400 analysis: The PM/400 software application summary includes an analysis of the top ten items for batch jobs, users, and communication lines. However, some jobs, users, or communication lines are not appropriate for such an analysis. For example, you may want to exclude jobs with longer than normal run times, such as autostart jobs, in the run-time category.

You can omit groups of batch jobs and users from the top ten analysis by using the generic omit function. For example, to omit all jobs starting with MYAPP specify: MYAPP*

To work with omissions, do the following steps:
1. Type **GO PM400** from the command line.
2. Type a 4 from the PM/400 Menu and press Enter. The Work with Top Ten Omissions display appears.
3. Type the appropriate option number depending on which item you want to omit
   - Type a 1 to work with jobs.
   - Type a 2 to work with users.
   - Type a 3 to work with communications lines.
4. Type a 1 in the appropriate field to omit either a user or a job from a particular category. In the case of a communications line, type the name of the line and then type a 1 in the appropriate field.

See Manage PM/400 to learn about other tasks that you can perform.

Turn off PM/400 momentarily: If you need to stop PM/400 from verifying that Collection Services is collecting data, you can use the scheduler job to change the date to a future date for the Q1PPMSUB job.
1. Type **GO PM400** from the command line.
2. Type a 2 (Work with automatically scheduled jobs).
3. Type a 2 (Change) next to the Q1PPMSUB job.
4. Change the date or time to a future date and time.
5. Press Enter. This change will momentarily stop PM/400 from verifying that Collection Services is collecting data. You must end what is currently being collected.

Note: PM/400 will not start, cycle, or change Collection Services until the date and time to which you set the Q1PPMSUB job has been reached.

See Schedule jobs with PM/400 to learn about other things that you can do with the scheduler.

See Manage PM/400 to learn about other tasks that you can perform.

Display PM/400 status: You can use either iSeries Navigator or the PM/400 Menu on your server to display the status of PM/400. Use the Performance Management/400 Status dialog to view the overall status of PM/400 on one or more servers or groups. For example, you are shown details as to whether PM/400 is active. Use the PM/400 Menu to view the Collection Services status, PM/400 scheduler status, the performance data release, the last transmission attempt, performance data members, and the performance data size.

To view the overall status for PM/400 from iSeries Navigator, do the following steps:
1. In iSeries Navigator, expand an endpoint system or a system group.
2. Expand **Configuration and Service**.
3. Right-click **Collection Services**.
4. Select **Performance Management/400**.
5. Select **Status**.

To view the detailed status for PM/400 from the PM/400 menu, do the following steps:
1. Type **GO PM400** from the command line.
2. Type a 6 from the command line and press Enter. See the online help for descriptions of each field.

See Manage PM/400 to learn about other tasks that you can perform.

View PM/400 reports: The output of the PM/400e offering is a set of management reports and graphs on a monthly or quarterly basis. The PM/400e offering has two options for the reports.

The purpose of the reports and graphs is to give management a clear understanding of the current performance of their servers and an accurate growth trend. To view each report and graph in detail and learn about some of their benefits and uses, see the PM/400 Web site

.

See Manage PM/400 to learn about other tasks that you can perform.

Performance Tools

The Performance Tools for iSeries licensed program allows you to analyze performance data in a variety of ways. Performance Tools is a collection of tools and commands for viewing, reporting, and graphing performance data. You can use Performance Tools for iSeries to view performance data collected with Collection Services or to view trace data collected with the Start Performance Trace (STRPFRTRC) command. You can then summarize the data into a report to research a performance problem on your system. You can also create graphs of the performance data to see resource utilization over time.

Performance Tools for iSeries contains a base product and two features (Manager and Agent). The base plus one of the features is required. For more information about the Manager and Agent features of Performance Tools, see the Manager and Agent feature comparison topic.

Performance Tools concepts
Describes a variety of tools to help you collect and analyze performance information. Find detailed information about exactly which tools perform which functions and how they work.

Install and configure Performance Tools
See this topic for installation and setup instructions.

Performance Tools reports
Performance Tools reports provide information on data that has been collected over time. Use these reports to get additional information about the performance and use of system resources.

For more detailed information about how to use Performance Tools to collect data about the performance of a system, job, or program, look in the Performance Tools book

. It also explains how to analyze and print data to help identify and correct any problems. See the HVLPTASK task to find out how Performance Tools shows the CPU time that is consumed by this task.

Performance Tools concepts

The Performance Tools for iSeries licensed program analyzes two distinct types of performance data: sample data and trace data. Collection Services collects sample data, which is summary data that is captured at regular time intervals. You collect sample data for trend analysis and performance analysis. The data relates to things such as storage pools and response times. However, Collection Services does not support the collection of trace data. Trace data is detailed data that you collect to gain additional information about specific jobs and transactions. To collect trace data, you can use either the Start Performance Trace (STRPFRTRC) command or performance explorer.

Functions included in Performance Tools
Performance Tools includes a variety of applications for collecting, analyzing, and reporting performance data. Knowing which functions are available, and which are best suited for a given task can be complex. See this topic for a description of the functions included in this licensed program.

Manager and Agent feature comparison
You can use the Manager and Agent features to efficiently divide required functions of Performance Tools over a distributed environment. This topic contains a description of these two features, the functions they each contain, and information about how to use them most effectively.

Displaying performance information
You can view system resource utilization data in iSeries Navigator. You can view the data, graph it, and summarize it into reports. Find information about how to access this function here.

Functions provided in Performance Tools: Performance Tools includes reports, interactive commands, and other functions. For example, Performance Tools includes the following:

Tool	Description
Work with System Activity (WRKSYSACT) command	The Work with System Activity (WRKSYSACT) command allows you to work interactively with the jobs, threads and tasks currently running in the system. The WRKSYSACT command reports system resource utilization, including CPU utilization on a per-task basis for partitions that use a shared processing pool. See the HVLPTASK task to find out how this command shows the CPU time that is consumed by this task.
Display Performance Data	The Display Performance Data graphical interface allows you to view performance data, summarize the data into reports, create graphs to show trends, and analyze the details of your system performance all from within iSeries Navigator.
Reports	The reports organize Collection Services performance data in a logical and useful format. This tool is discussed in detail in the Performance Tools book .
Graphics function	The Performance Tools graphics function allows you to work with performance data in a graphical format. You can display the graphs interactively, or you can print, plot, or save the data to a graphics data format (GDF) file for use by other utilities. This tool is discussed in detail in the Performance Tools book .
Performance explorer	Performance explorer is a data collection tool that helps you identify the causes of performance problems that cannot be identified by sample data that was collected by Collection Services or by doing general trend analysis. Use performance explorer for detailed application analysis at a program, procedure, module, or method level. For example, you can collect trace data on an individual program or procedure CPU and I/O statistics or individual object I/O characteristics. This tool is discussed in detail in the Performance Tools book .

Manager and Agent feature comparison: Performance Tools is available with two separately installable features. This topic explains the differences between the two features to help you decide which feature is more appropriate for your applications.

Manager feature
The Performance Tools Manager feature is a full-function package, intended to be used on the central site system in a distributed environment or on a single system. If you require analysis of trace data, viewing data graphically, viewing system activity in real time, or managing and tracking system growth, the Manager feature of the Performance Tools licensed program is more useful.

Agent feature
The Performance Tools Agent feature, with a subset of the Manager function, is a lower-priced package

with the more basic functions. In a distributed environment, the Agent feature works well for managed systems in the network because the data can be sent to the Manager if detailed analysis is required. It is also an effective tool for sites that need a reasonable level of self-sufficiency but have no expert skills available.

The Agent feature of Performance Tools provides functions to simplify the collection, management, online display, data reduction, and analysis of performance data. The Performance explorer reporting function and its associated commands are included in the base option in the Performance Tools for iSeries licensed program and, therefore, are available with either the Manager feature or the Agent feature. The major Performance Tools functions not contained in the Agent feature are performance and trace reports, performance utilities (job traces and the select file and access group), system activity monitoring, and performance graphics.

Display performance information: Performance Tools can display performance data from iSeries Navigator. From this graphical interface, you can view performance data, summarize the data into reports, create graphs to show trends, and analyze the details of your system performance.

Metrics
iSeries Navigator displays performance metrics over a selected interval of time. The performance metrics you can view in the Graphs pane of the Display Performance Data window include:
- Transaction Count
- Transaction Response Time
- Total CPU Utilization
- Interactive CPU Utilization
- Batch CPU Utilization
- High Disk Utilization
- Machine Pool Page Faults/Second
- User Pool Page Faults/Second
- Exceptions

The Details pane allows you to view detailed performance data for the selected time interval in a variety of ways. To analyze your system performance, you can view job data, subsystem data, pool data, or disk unit data.

Reports
In addition to viewing graphs and detail data, you can also print reports from the Display Performance Data window. Performance reports allow you to research areas of the system that are causing performance problems. You can run different reports to see where your system resources are being used. Printing reports in Performance Tools is available only when option 1 (Manager feature) of Performance Tools for iSeries (5722-PT1) is installed on the central system. For more information on the Manager feature, see the Manager and Agent feature comparison topic.

The reports you can print from the Display Performance Data window include:
- System
- Component
- Job
- Pool
- Resource

Accessing through iSeries Navigator
The Display Performance Data window can be accessed in iSeries Navigator by following these steps:
1. In iSeries Navigator, expand **My Connections** (or your active environment).

2. Expand the server that contains the performance data you want to view.
3. Expand **Configuration and Service**.
4. Right-click **Collection Services**, select **Performance Tools**, and then select **Performance Data**.
5. Select the performance data file that you want to display.
6. Click **Display**.

For more information on how to use the Display Performance Data window in iSeries Navigator, see the iSeries Navigator online help.

Install and configure Performance Tools

To install Performance Tools, you need a user profile with save system (*SAVSYS) authority. You can use the system operator profile to obtain this authority.

Performance Tools must run in a library named QPFR. If you have a library by this name on your system, use the Rename Object (RNMOBJ) command to rename it before you install Performance Tools. This step will ensure the proper operation of Performance Tools.

Use the following command to place the Performance Tools in library QPFR:

```
RSTLICPGM LICPGM(5722PT1) DEV(NAME) OPTION(*BASE)
```

You must then perform one of the following:

- If you have purchased the Manager feature, use the following command:
  ```
  RSTLICPGM LICPGM(5722PT1) DEV(tape-device-name) OPTION(1)
  ```
- If you have purchased the Agent feature, use the following command:
  ```
  RSTLICPGM LICPGM(5722PT1) DEV(NAME) OPTION(2)
  ```

If you have several CD-ROMs to install, the following situation may occur. After installing the first one, you may receive a message saying that the licensed program is restored but no language objects were restored. If this occurs, insert the next CD-ROM and enter the following:

```
RSTLICPGM LICPGM(5722PT1) DEV(NAME) RSTOBJ(*LNG) OPTION(*BASE)
```

Another method for installing the Performance Tools program is to type GO LICPGM and use the menu options.

Performance Tools is a processor-based program. The usage type is concurrent, and the program is installed with a usage limit *NOMAX.

This program is discussed in detail in the Performance Tools book

.

Performance Tools reports

Performance Tools provides an easy way for you to look at your collected data. Collection Services provides data for most of the Performance Tools reports with the exception of the Transaction, Lock, and Trace reports. You must use the STRPFRTRC and ENDPFRTRC commands to collect the trace information for those three reports. With Performance Tools reports, you can isolate performance problems. After you have collected performance data over time, you can print reports to see how and where system resources are being used. The reports can direct you to specific application programs, users, or inefficient workloads that are causing slower overall response times.

The following list describes each report and gives a brief overview as to why you would use a particular report. Each report is discussed in detail in the Performance Tools book

.

Note: You can print reports for data that was collected in previous releases with the Start Performance Monitor (STRPFRMON) command.

Overview of Performance Tools reports

Report	Description	What is shown	How you use the information
System	Uses Collection Services data to provide an overview of how the system is operating. The report contains summary information on the workload, resource use, storage pool utilization, disk utilization, and communications. Run and print this report often to give you a general idea of your system use.	System workload. The report includes the database capabilities data.	Workload projection
Component	Uses Collection Services data to provide information about the same components of system performance as a System Report, but at a greater level of detail. This report helps you find which jobs are consuming high amounts of system resources, such as CPU, disk, and so on.	Resource use, communications, system and user jobs. The report includes the database capabilities data and the Interactive Feature utilization.	Hardware growth and configuration processing trends
Transaction	Uses trace data to provide detailed information about the transactions that occurred during the performance data collection.	Workload and utilization of CPU, disk, main storage, transaction workload, object contention	Workload projection, pool configuration, application design, file contention, and program use
Lock	Uses trace data to provide information about lock and seize conflicts during system operation. With this information you can determine if jobs are being delayed during processing because of unsatisfied lock requests or internal machine seize conflicts. These conditions are also called waits. If they are occurring, you can determine which objects the jobs are waiting for and the length of the wait.	File, record, or object contention by time; the holding job or object name; the requesting job or object name	Problem analysis. Reduction or elimination of object contention.

Trace	Uses trace data to show the progression of different job types (for example, batch jobs) traced through time. Resources utilized, exceptions, and state transitions are reported.	Job class time-slice end and trace data	Problem analysis and batch job progress
Job	Uses Collection Services data to show information on all or selected intervals and jobs, including detail and summary information for interactive jobs and for noninteractive jobs. Because the report can be long, you may want to limit the output by selecting the intervals and jobs you want to include.	Jobs by interval	Job data
Pool	Uses Collection Services data to provide a section on subsystem activity and a section on pool activity. Data is shown for each sample interval. Because the report can be long, you may want to limit the output by selecting the intervals and jobs you want to include.	Pools by interval	Pool data
Resource interval	Uses Collection Services data to provide resource information on all or selected intervals. Because the report can be long, you may want to limit the output by selecting the intervals you want to include.	Resources by interval	System resource use

Performance explorer and Collection Services are separate collecting agents. Each one produces its own set of database files that contain grouped sets of collected data. You can run both collections at the same time.

For a list of reports for other tools, see the following:
- Performance explorer reports
- Performance Management/400 reports

Performance explorer

Performance explorer is a data collection tool that helps the user identify the causes of performance problems that cannot be identified by collecting data using Collection Services or by doing general trend analysis. Two reasons to use performance explorer include:

- Isolating performance problems to the system resource, application, program, procedure, or method that is causing the problem
- Analyzing the performance of applications

The collection functions and related commands of performance explorer are part of the OS/400 licensed program. The reporting function and its associated commands are part of the base option in the Performance Tools for iSeries licensed program and therefore, are available with either the Manager feature or the Agent feature. The AS/400 Performance Explorer Tips and Techniques book

provides additional examples of the performance explorer functions and examples of the enhanced performance explorer trace support.

Performance explorer is a tool that helps find the causes of performance problems that cannot be identified by using tools that do general performance monitoring. As your computer environment grows both in size and in complexity, it is reasonable for your performance analysis to gain in complexity as well. The performance explorer addresses this growth in complexity by gathering data on complex performance problems.

Note: Performance explorer is the tool you need to use after you have tried the other tools. It gathers specific forms of data that can more easily isolate the factors involved in a performance problem; however, when you collect this data, you can significantly affect the performance of your system.

This tool is designed for application developers who are interested in understanding or improving the performance of their programs. It is also useful for users knowledgeable in performance management to help identify and isolate complex performance problems.

To learn more about performance explorer, refer to any of the following performance explorer topics.

Performance explorer concepts
Performance explorer works by collecting detailed information about a specified system process or resource. This topic explains how performance explorer works, and how best to use it.

Configure performance explorer
To collect detailed trace information, you need to tailor performance explorer to work optimally with the application process from which the trace is being taken.

Performance Explorer reports
After you have collected performance data with a performance explorer session, you can view it by running the included reports or by querying the database files directly.

For more detailed information, refer to the Performance Tools book

.

Performance explorer concepts

Like Collection Services, performance explorer collects data for later analysis. However, they collect very different types of data. Collection Services collects a broad range of system data at regularly schedules intervals, with minimal system resource consumption. In contrast, performance explorer starts a session that collects trace-level data. This trace generates a large amount of detailed information about about the resources consumed by an application, job, or thread. Specifically, you can use Performance Explorer to answer specific questions about areas like system-generated disk I/O, procedure calls, Java method calls, page faults, and other trace events

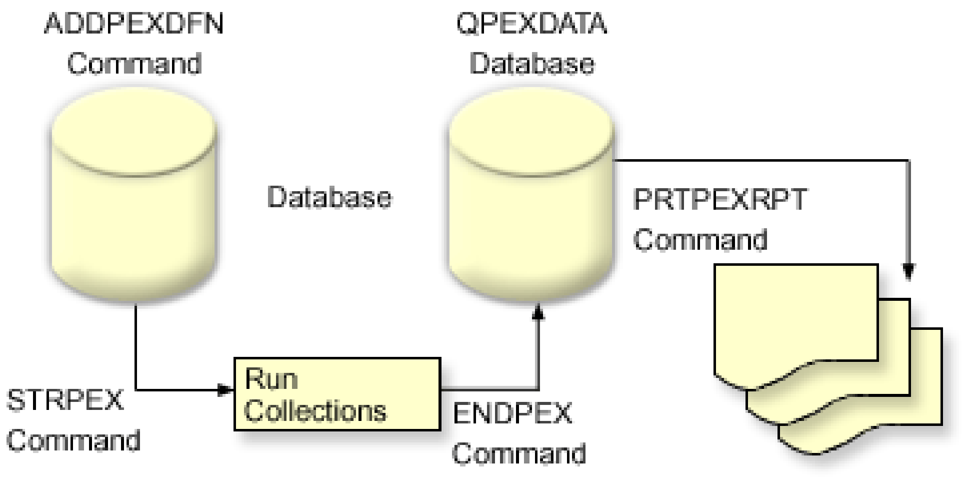
. It is the ability to collect very specific and very detailed information that makes the performance explorer effective in helping isolate performance problems. For example, Collection Services can tell you that disk storage space is rapidly being consumed. You can use performance explorer to identify what programs and objects are consuming too much disk space, and why.

Note: You can collect performance explorer data and Collections Services data at the same time.

How performance explorer works

The following figure should help you become familiar with the normal path through the performance explorer. For details on each of these steps, see Configure performance explorer. The figure shows a basic work cycle that consists of the following steps:

1. Define a performance explorer data collection. You can also add a filter to limit the amount of data collected by specifying a compare value for specific events.
2. Start the performance explorer to collect the data based on your definition.
3. Run your program, command, or workload.
4. End the collection, which saves the collected data to a set of database files.
5. Create and print reports from the database files.



To learn more about performance explorer, refer to any of the following performance explorer topics.

Performance explorer definitions
The parameters and conditions that determine what data performance explorer collects and how it collects it are configured and stored using performance explorer definitions. This topic explains how to use these definitions and provides a sample illustrating a simple definition.

Performance explorer database files
The data that performance explorer collects is stored in performance explorer database files.

Performance explorer benefits
Performance explorer contains a variety of functions that can help gather and analyze detailed performance information. This topic provides an overview of those various functions.

Performance explorer definitions: To collect performance explorer data, you need to tell performance explorer what data to gather. You do this by using the Add Performance Explorer Definition (ADDPEXDFN) command to create a performance explorer definition. After the definition is completed and saved, you are ready to continue to the next task in the cycle of work.

Before creating a new definition, consider what kinds of information you want and the amount of detail you need. The performance explorer provides the following types of data collection:

Statistics type definitions
Identifies applications and IBM programs or modules that consume excessive CPU use or that perform a high number of disk I/O operations. Typically, you use the statistical type to identify programs that should be investigated further as potential performance bottlenecks.

- Good for first order analysis of OS/400 programs, procedures, and MI complex instructions.
  - Gives number of invocations
  - Gives both inline and cumulative CPU usage in microseconds
  - Gives both inline and cumulative number of synchronous and asynchronous I/O
  - Gives number of calls made
- Works well for short or long runs
- Size of the collected data is fairly small and constant for all runs
- Run time collection overhead of ILE procedures may be a problem due to the frequency of calls. Although run time is degraded, the collected statistics are still accurate because Performance Explorer removes most of the collection overhead from the data.
- Uses combined or separated data areas. The MRGJOB parameter on the ADDPEXDFN command specifies whether all program statistics are accumulated in one data area, or kept separate (for example, one data area for each job).

The statistics can be structured in either a hierarchical or flattened manner.

- A hierarchical structure organizes the statistics into a call tree form in which each node in the tree represents a program procedure run by the job or task.
- A flattened structure organizes the statistics into a simple list of programs or procedures, each with its own set of statistics.

Here is an example of a performance explorer statistics definition called MYSTATS that will show CPU and disk resource usage on a per program or procedure level.

```
ADDPEXDFN DFN(MYSTATS) /* The name of the definition. */
TYPE(*STATS) /* The type of definition */
JOB(*ALL) /*All Jobs */
TASKS(*ALL) /*All tasks */
MRGJOB(*YES) /* Merge records to reduce collection overhead */
DTAORG(*FLAT) /* Do not keep track of who calls who */
```

Profile type definitions
Identifies high-level language (HLL) programs that consume excessive CPU utilization based on source program statement numbers. You can also identify a program that is constantly branching between the start of the program and subroutines at the end of the program. If the program is large enough, this constant jumping back and forth can cause excessive page fault rates on a system with limited main storage.

- Program profile (specify TYPE(*PROFILE) and PRFTYPE(*PGM) on the ADDPEXDFN command)
  - Gives detailed breakdown of where you are spending time within a set of programs within a specific job.
  - Can summarize the data by program, module, procedure, statement, or instruction.
  - Size of collection is fairly small and constant regardless of length of run.
  - Limit of 16 MI programs means that you should use this as a second order analysis tool.

  - Can vary overhead by changing sample interval. An interval of 2 milliseconds seems a good first choice for benchmarks.
  - No restrictions on pane size due to the number of programs specified or the size of the programs specified.

  Here is an example of a performance explorer program profile definition called PGMPROF that will show usage for a particular procedure.

```
ADDPEXDFN DFN(PGMPROF) /* The name of the definition. */
TYPE(*PROFILE) /* The type of definition */
JOB(*ALL) /*All Jobs */
PGM((MYLIB/MYPGM MYMODULE MYPROCEDURE)) /* The name of the program to monitor. */
INTERVAL(1) /* A 1-millisecond sample will be taken. */
```

- Job profile (specify the following on the ADDPEXDFN command: TYPE(*PROFILE) and PRFTYPE(*JOB))
  - Gives detailed breakdown of where you are spending time in the set of jobs or tasks of the collection.
  - Size of collection is relatively small but not constant. The size increases as the length of the run increases.
  - Can profile all jobs and tasks on the system or can narrow the scope of data collected to just a few jobs or tasks of interest.
  - Can vary overhead by changing sample interval. An interval of 2 milliseconds seems a good first choice for benchmarks.

  Here is an example of a performance explorer job profile definition called ALLJOBPROF that will show usage for all your jobs.

```
ADDPEXDFN DFN(ALLJOBPROF) /* The name of the definition. */
TYPE(*PROFILE) /* The type of definition */
PRFTYPE(*JOB) /* A job profile type will be monitored. */
JOB(*ALL) /*All Jobs */
TASKS(*ALL) /*All tasks */
INTERVAL(1) /* A 1-millisecond sample will be taken. */
```

Trace definitions
Gathers a historical trace of performance activity generated by one or more jobs on the system. The trace type gathers specific information about when and in what order events occurred. The trace type collects detailed reference information about programs, Licensed Internal Code (LIC) tasks, OS/400 job, and object reference information.

- Some common trace events are:
  - Program and procedure calls and returns
  - Storage, for example, allocate and deallocate.
  - Disk I/O, for example, read operations and write operations.
  - Java method, for example, entry and exit.
  - Java, for example, object create and garbage collection.
  - Journal, for example, start commit and end commit.
  - Synchronization, for example, mutex lock and unlock or semaphore waits.
  - Communications, for example, TCP, IP, or UDP.
- Longer runs collect more data.

Here is an example of a performance explorer trace definition called DISKTRACE that will show usage for all disk events.

```
ADDPEXDFN DFN(DISKTRACE) /* The name of the definition. */
TYPE(*TRACE) /* The type of definition */
JOB(*ALL) /*All Jobs */
```

```
 TASKS(*ALL) /*All tasks */
 TRCTYPE(*SLTEVT) /* Only selected individual events and machine instructions
are included in the trace definition */
 SLTEVT(*YES) /* *SLTEVT allows you to specify individual machine instructions
and events to be specified in addition to the categories of events
available with the TRCTYPE parameter. */
 DSKEVT((*ALL)) /* All disk events are to be traced. */
```

Performance explorer database files: The following table shows the performance explorer data files collected by the system when using data collection commands. Type the Display File Field Description (DSPFFD) command as follows to view the contents for a single file:

```
DSPFFD FILE(xxxxxxxxx)
```

where *xxxxxxxxx* is the name of the file that you want to display.

Type of information contained in file	File name
Reference information	QAYPEREF
General information	QAYPERUNI
PMC selection	QAYPEFQCFG
Basic configuration information	QAYPECFGI
Machine interface (MI) complex instructions collected on	QAYPELCPLX
Jobs collected on	QAYPELJOB
Metrics to collect data on	QAYPELMET
Machine interface (MI) program, module, or procedures collected on	QAYPELMI
Licensed Internal Code (LIC) modules to collect data on	QAYPELLIC
Task names to collect data on	QAYPELNAMT
Task number to collect data on	QAYPELNUMT
Machine interface (MI) complex instructions mapping	QAYPEMICPX
Event type and subtype mapping	QAYPEEVENT
Hardware mapping data	QAYPEHWMAP
Licensed Internal Code (LIC) address resolution mapping	QAYPEPROCI
Segment address resolution mapping	QAYPESEGI
Process and task resolution mapping	QAYPETASKI
Common trace data for all events	QAYPETIDX
Auxiliary storage management event data	QAYPEASM
Base event data	QAYPEBASE
Disk event data	QAYPEDASD
Disk server event data	QAYPEDSRV
Page fault event data	QAYPEPGFLT
Resource management process event data	QAYPERMPM
Resource management seize lock event data	QAYPERMSL
Advanced 36 event data	QAYPES36
Segment address range (SAR) data	QAYPESAR
Unknown event data	QAYPEUNKWN
Basic statistics data	QAYPESTATS
Statistic profiling summary data	QAYPEPSUM

Type of information contained in file	File name
Licensed Internal Code (LIC) bracketing data	QAYPELBRKT
Machine interface (MI) user event data	QAYPEMIUSR
Machine interface (MI) program bracketing data	QAYPEMBRKT
Addresses of machine interface (MI) pointer	QAYPEMIPTR
User-defined bracketing hook data	QAYPEUSRDF
Hardware monitor data	QAYPEHMON
Hardware monitor total data	QAYPEHTOT
Release, version, modification level	QRLVRM
Performance explorer level indicator	QRLLVL
Performance explorer Java event data	QAYPEJVA
Performance explorer Java class information data	QAYPEJVCI
Performance explorer Java method information data	QAYPEJVMI
Performance explorer Java name information data	QAYPEJVNI
Synchronization event data	QAYPESYNC
Communications event data	QAYPECMN
File Serving event data	QAYPEFILSV
Heap event data	QAYPEHEAP
PASE event data	QAYEPASE
Trace job equivalent event data	QAYPETBRKT
Task switch event data	QAYPETSKSW
Synchronization event data	QAYPESYNC
Program profile data	QAYPEPPANE

Performance explorer reports: Performance explorer gathers detailed information about a program or job's behavior and performance and stores this information in performance explorer database files. You can query these files with SQL, or by running one of several reports. You can generate four different reports with performance explorer: Statistics, Profile, Trace, and Base reports. See Performance explorer definitions for information on why you would use a particular definition to generate one of these reports. Each report is discussed in detail in the Performance Tools book

.

You can create and print performance explorer reports by using the Print Performance Explorer Report (PRTPEXRPT) command. Use the OUTFILE parameter when you want to customize your Trace Report. The following commands are examples for printing reports for each type of performance explorer data:

- Print a *STATS report sorting by the CPU time used

```
PRTPEXRPT MBR(MYSTATS) LIB(MYLIB) TYPE(*STATS) STATSOPT(*CPU)
```

- Print a profile report summarized by procedure

```
 PRTPEXRPT MBR(MYPROFILE) LIB(MYLIB) TYPE(*PROFILE) PROFILEOPT(*SAMPLECOUNT *PROCEDURE)
```

- Print a trace sorted by task ID

```
PRTPEXRPT MBR(MYTRACE) LIB(MYLIB) TYPE(*TRACE) TRACEOPT(*TASK)
```

Performance explorer stores its collected data in the QAVPETRCI file, which is located in the QPFR library. Type the following command to view the contents for a single record:

```
DSPFFD FILE(QPFR/QAVPETRCI)
```

Performance explorer benefits: Performance explorer has advantages for people who need detailed performance analysis on an iSeries server. Using performance explorer you can:

- Determine what is causing a performance problem on the system down to the level of user, job, file, object, thread, task, program, module, procedure, statement, or instruction address.
- Collect performance information on user-developed and system software.
- Do a detailed analysis on one job without affecting the performance of other operations on the system.
- Analyze data on a system other than the one on which it was collected. For example, if you collect data on a managed system in your network, you can send it to the central site system for analysis.

Configure performance explorer

To configure performance explorer, follow these steps:

1. Create a session definition that informs the iSeries server which performance data you want to collect. On the Add Performance Explorer Definition (ADDPEXDFN) display, specify the collection type and a name for the definition. This definition is stored as a database member by that name in the QAPEXDFN file in library QUSRSYS. The name that you specify is used on the Start Performance Explorer (STRPEX) command.
2. Add a filter (ADDPEXFTR command). A performance explorer filter identifies the performance data that is to be collected during a performance explorer session, and is meant to limit the amount of data collected by specifying a compare value for specific events.
3. Start collecting data (STRPEX command). A job may be in more than one performance explorer collection if the *PMCO event is not being collected. If the *PMCO event is being collected, then a job can be in more than one collection only if all the collections have the same interval specification (ADDPEXDFN INTERVAL() parameter).
4. Run your command, program, or workload for data that you want to analyze.
5. Stop collecting the data and save it to database files for analysis. Use the End Performance Explorer (ENDPEX) command to stop the collection.
6. Analyze the performance data. The Print Performance Explorer Report (PRTPEXRPT) command, included in the Performance Tools licensed program, provides unique reports for each type of data (statistical, profile, trace profile, or trace). The other option for analysis is to write your own queries over the set of database files.

All of the performance explorer commands can be accessed with one of the following methods:

- The command interface. Type the commands from the command line. All the commands are part of the OS/400 operating system, except the Print Performance Explorer Report (PRTPEXRPT) command.
- The Performance Tools menu options.

To see a performance explorer work cycle, see Performance explorer concepts.

Ending performance explorer

To end the performance explorer session, use the End Performance Explorer (ENDPEX) command. The ENDPEX command performs the following actions on the collected data:

- Places the collected data in files QAYPExxx in the library that you specify.
  Use OPTION(*END) and DTAOPT(*LIB) to do this. The database member name for all the QAYPExxx files uses the session name as the default unless you specify a name for the DTAMBR parameter. You can specify RPLDTA(*NO) to erase data that was collected using this session name or RPLDTA(*YES) to add the collected data to the existing data. Unless you are a very sophisticated user, use RPLDTA(*NO).
- Places the collected data into a single IBM-defined file.
  Use OPTION(*END) and DTAOPT(*MGTCOL) to do this. Typically, you would use *MGTCOL only under

the direction of an IBM service representative. Specifying the *MGTCOL value on the DTAOPT parameter saves the collection information into a management collection object. The management collection object option should be used only if the data is going to be shipped to IBM. The performance tools can analyze only the database files.

- Discards the collected data.
  Use OPTION(*END) if you want to save the data or DTAOPT(*DLT) to discard any collected data. You do this when you determine the collected data cannot be used. For example, one of the suspected jobs did not start as expected. If you choose the *DLT option, the collected performance data for the session is never saved.
- Suspends the collection session but does not end it.
  Use OPTION(*SUSPEND) to do this. You can later start the data collection again by issuing the STRPEX command with OPTION(*RESUME) for the specific session ID.

Note: If you forget the active collection session name, use the ENDPEX SSNID(*SELECT) command.

iDoctor for iSeries

iDoctor for iSeries is a suite of tools consisting of three components: Job Watcher, Object Explorer, and the performance explorer Analyzer.

Job Watcher and the performance explorer Analyzer are for performance analysis. The server-side portion of these components consists of a variety of data collection and analysis programs designed to consolidate performance data in a more usable format. The client-side components for Job Watcher and the Analyzer consist of graphical interfaces for displaying the server data in flexible graph and table views.

Object Explorer is a tool designed to make the iSeries data more easily accessible with a graphical interface. Any object type on the system can be listed, viewed, and described. Object Explorer allows you to access any physical files or logical files on the system and create your own queries and graphs over the data with the easy-to-use Query Definition and Graph Definition interfaces.

Job Watcher

Job Watcher displays real-time tables and graphical data that represent, in a very detailed way, what a job is doing and why it is not running. Job Watcher provides several different reports that provide detailed job statistics by interval. These statistics allow you to determine things like CPU utilization, DASD counters, waits, faults, call stack information, conflict information, and more.

Object Explorer

Object Explorer allows you to browse and work with the objects on an iSeries server. The primary feature of Object Explorer is the Data Viewer, which lets you display the contents of any physical file on the system and write your own queries and graphs over the data. Also included in Object Explorer are the following features:

- Easy, fast filtering of lists of libraries and objects
- The ability to copy (CRTDUPOBJ), cut (MOVOBJ), rename (RENOBJ), and delete (DLT*) objects through a fast, easy-to-use user interface
- Browse logical or physical file members
- Define your own queries and save them to use later with the Query Definition interface
- Define your own graphs and save them to use later with the Graph Definition interface

Performance explorer Analyzer

Performance explorer Analyzer evaluates the overall performance of your system and builds on what you have done with the Performance Tools licensed program. The Analyzer condenses volumes of trace data into reports that can be graphed or viewed to help isolate performance issues and reduce overall problem determination time. The Analyzer provides an easy-to-use graphical interface for analyzing CPU utilization, physical disk operations, logical disk input/output, data areas, and data queues. The Analyzer can also help you isolate the cause of application slowdowns.

Visit the iDoctor for iSeries

Web site for more information.

Performance Trace Data Visualizer (PTDV)

The Performance Trace Data Visualizer (PTDV) is a Java application that can be used for performance analysis of applications running on iSeries servers. PTDV works with performance explorer in the OS/400 base operating system to allow the analyst to view program flows and get details (such as CPU time, current system time, number of cycles, and number of instructions) summarized by trace, job, thread, and procedures. When visualizing Java application traces, additional details such as the number and type of objects created and information about Java locking behavior can be displayed. There is also support for performance explorer events generated by the WebSphere Application Server. PTDV allows sorting of columns, exporting of data, and many levels of data summarization.

For more information, go to the Performance Trace Data Visualizer

Web site.

Performance Management APIs

The performance management APIs allow you to collect and manage performance data using Collection Services, performance collector, performance explorer, and Performance Management/400 (PM/400).

The Performance Management APIs include:

- Collection Services APIs
- Performance Collector APIs
- Performance Explorer (PEX) APIs
- Performance Management/400 (PM/400) APIs

“Work with” commands for OS/400 performance

OS/400 includes a number of commands that can allow you to perform real-time monitoring of performance data from the character-based interface. You can use these commands to answer specific questions about system performance and to help you tune your system. For information about real-time monitoring from iSeries Navigator, see iSeries Navigator monitors.

Command	Function
Work with Active Jobs (WRKACTJOB)	Allows you to review and change the attributes and resource utilization of the jobs running on your system.
Work with Disk Status (WRKDSKSTS)	Display the performance information and attributes for system disk units.
Work with System Status (WRKSYSSTS)	Provides an overview of current system activity. Specifically, it displays the number of jobs on the system and storage pool utilization information.
Work with System Activity (WRKSYSACT)	Work with jobs and tasks on your system. This command is part of the Performance Tools licensed program (PT1).
Work with Object Locks (WRKOBJLCK)	Work with and display locks on a specified object, including locks waiting to be applied.
Work with Shared Storage Pools (WRKSHRPOOL)	Display the utilization information and change attributes of shared storage pools, including machine and base pool.

Extended Adaptive Cache

Note: Feature Codes #4331 and #6831 (CCIN #6731) have been withdrawn from marketing. The information provided here is a reference for existing users. Any future enhancements to this cache feature of iSeries storage I/O will be available through these pages.

Improve your iSeries system performance with Extended Adaptive Cache! Extended Adaptive Cache is an advanced large read cache technology that improves both the I/O subsystem and system response times by reducing the number of physical I/O requests that are read from disk. Extended Adaptive Cache generates statistical information for the data and then uses a mix of management strategies to determine which data to cache. Extended Adaptive Cache has proven to be highly effective on many types of workloads.

IBM offers an innovative tool that allows you to determine the benefits that Extended Adaptive Cache can provide in your iSeries computing environment. **Extended Adaptive Cache Simulator** is activated through the Collection Services function within iSeries Navigator. The Simulator shows emulated performance results for an actual workload over time on a per disk basis. Extended Adaptive Cache Simulator performs at the I/O storage adapter level, and uses the same algorithms that manage the Extended Adaptive Cache.

To learn more, keep reading:

- **Extended Adaptive Cache concepts**
  Explore Extended Adaptive Cache. Find information about planning, restrictions, and important considerations before you begin to use this tool.
- **Extended Adaptive Cache Simulator**
  Learn how to use Extended Adaptive Cache to determine the response time improvements that Extended Adaptive Cache can provide in your computing environment.
- **Get Extended Adaptive Cache**
  After you have used the Extended Adaptive Cache Simulator to see the benefits this tool can offer to your environment, learn about how to get Extended Adaptive Cache.

Extended Adaptive Cache Concepts

Improve system performance with Extended Adaptive Cache, an advanced read cache technology that improves both the I/O subsystem and system response times by reducing the number of physical I/O requests that are read from disk. Extended Adaptive Cache not only improves the performance of database-read actions, but of all read actions. This includes read actions that are generated by other system components such as the Integrated xSeries Server. It also works effectively in storage subsystems that have device parity protection or mirrored protection. Extended Adaptive Cache has proven to be highly effective on many types of workloads.

How the Extended Adaptive Cache works

Extended Adaptive Cache is integrated into the iSeries I/O subsystem. It operates at the disk subsystem controller level and does not affect the iSeries system processor. The storage I/O adapter manages the Extended Adaptive Cache by using a Read Cache Device (such as a solid state disk) to provide the cache memory.

Extended Adaptive Cache generates statistical information for the data, and then uses a mix of management strategies to determine which data to cache. The management of the cache is performed automatically within the I/O adapter and is designed to cache data by using a predictive algorithm. The algorithm considers how recently and how frequently the host has accessed a predetermined range of data.

The design of Extended Adaptive Cache was based on specific data management strategies of the iSeries server. Whether the disks are device parity protected, mirrored, or unprotected, the data stored on the disks has a tendency to occur in bands. This means that there are physically contiguous areas of disk storage where data is actively read, physically contiguous areas that are frequently written to, physically contiguous areas that are both actively read and written to, or physically contiguous areas of storage that are not frequently accessed.

This "banding" of data is accounted for in the Extended Adaptive Cache design. The goal is to cache bands characterized as read/write and read-only. A band that is characterized as write-only, while cached in the storage subsystem write cache, remains largely unaffected by Extended Adaptive Cache. Extended Adaptive Cache is also designed to not harm the performance of large blocks of data that are either sequentially written or sequentially read. In this instance, the pre-fetch capability of the disks, as well as other caches in the system, ensures a quick response time.

To learn more, keep reading:

- **Restrictions and considerations for Extended Adaptive Cache**
  See what components Extended Adaptive Cache requires and learn more about what to expect.
- **Extended Adaptive Cache Simulator**
  Learn how to use Extended Adaptive Cache Simulator to determine the response time improvements that Extended Adaptive Cache can provide in your computing environment.
- **Start Extended Adaptive Cache**
  Learn how to activate Extended Adaptive Cache.

Restrictions and considerations for Extended Adaptive Cache: Before you begin using Extended Adaptive Cache, you should do some initial planning to take into account any restrictions or considerations that may pertain to your computing environment.

Restrictions

To use Extended Adaptive Cache, your system must have the following:

- One or more storage I/O adapters that support Extended Adaptive Cache (CCIN 2748 for systems running V4R4 or later, CCIN 2778 for systems running V5R1 or later, or CCIN 2757 for systems running the latest release of V5R2 (see Information APAR II13365))
- A Read Cache Device (RCD) for each storage I/O adapter that Extended Adaptive Cache is to be activated on (CCIN 6731 for systems running V4R4 or later)
- Performance Tools for iSeries licensed program

Extended Adaptive Cache is automatically enabled through the RCD. There is no controlled on or off switch. The RCD may be added without system interruption through concurrent maintenance. The RCD resides in an internal disk slot and works with all other disk types and capacities. Be aware that all data in the Extended Adaptive Cache is also guaranteed to be on the disks. In the unlikely event of an RCD failure, there will be no data loss.

There are no restrictions for using Extended Adaptive Cache with regard to device parity protection and mirrored protection for other disks under the I/O adapter. However, Extended Adaptive Cache cannot be used in conjunction with Integrated Hardware Disk Compression on the same I/O adapter. Finally, Extended Adaptive Cache is designed specifically to complement iSeries Expert Cache, and may be used with or without it.

Considerations

Using the Extended Adaptive Cache allows you to attain a significant decrease in I/O response time and increase in system I/O throughput in most environments. As is the general case with caches, the system configuration and workload influence the effectiveness of Extended Adaptive Cache. Extended Adaptive Cache performs at the storage subsystem level. It caches data for the set of disks that are within that specific subsystem. Therefore, it is logical to add Extended Adaptive Cache to the most active and performance-critical storage subsystems within the system. Extended Adaptive Cache is not considered a pre-fetch type cache and therefore will not interfere with the read-ahead capabilities in the disk.

The larger the area of disk storage that is actively receiving I/O requests, the more selective Extended Adaptive Cache is about deciding when to bring new data into cache. This adaptive ability allows Extended Adaptive Cache to be effective on many workload types and sizes. The overall cache effectiveness is best understood from this perspective through the use of Extended Adaptive Cache Simulator.

Additionally, Extended Cache Simulator and Extended Adaptive Cache cannot be active at the same time on the same storage I/O adapter.

Once you are aware of these restrictions and considerations, you are ready to Start Extended Adaptive Cache.

Start Extended Adaptive Cache: To start Extended Adaptive Cache and increase your system's performance, purchase the Read Cache Device. Once the Read Cache Device has been inserted into a disk slot on the subsystem, Extended Adaptive Cache will be activated. There is no user-controlled on or off switch. It takes approximately an hour for Extended Adaptive Cache to monitor the data flow and populate the Read Cache Device. After an hour of running Extended Adaptive Cache, your system should show improved performance (depending on your current workload) and increased I/O throughput.

To find out whether your iSeries system is capable of using Extended Adaptive Cache, see Restrictions and considerations for Extended Adaptive Cache.

Extended Adaptive Cache Simulator

Note: Feature Codes #4331 and #6831 (CCIN #6731) have been withdrawn from marketing. The information provided here is a reference for existing users. Any future enhancements to this cache feature of iSeries storage I/O will be available through these pages.

Use Extended Adaptive Cache Simulator to estimate the system performance improvements with Extended Adaptive Cache. The Extended Adaptive Cache Simulator is a performance tool that can determine the response time improvements that Extended Adaptive Cache can provide on your system. This determination can be based on your system configuration and data workload, and is made before you purchase a Read Cache Device.

Extended Adaptive Cache Simulator is controlled within Collection Services and is available on systems running the latest release of V5R2 with CCIN 2757 I/O adapters. (The CCIN 2748, CCIN 2778, and CCIN 2757 I/O adapters are the same storage I/O adapters that support Extended Adaptive Cache itself.) The Simulator is flexible, allowing you to emulate different cache capacities to better determine the capacity that would best suit your specific system and workload needs. The actual Read Cache Device capacity is 1600 MB.

The performance information gathered through activation of Extended Adaptive Cache Simulator will give you an indication of the number of disk read operations that could be saved through the use of Extended Adaptive Cache. The performance data reflects potential improvements in disk access time.

Get Extended Adaptive Cache

After obtaining the performance data from Extended Adaptive Cache Simulator and deciding that you want Extended Adaptive Cache to improve your system's performance, you must purchase a Read Cache Device (RCD). Extended Adaptive Cache is automatically enabled through the RCD.

To begin using Extended Adaptive Cache, you must have:

- One or more storage I/O adapters that support Extended Adaptive Cache (CCIN 2748 for systems running V4R4 or later, or CCIN 2778 for systems running V5R1 or later, or CCIN 2757 for systems running V5R2HW).
- A Read Cache Device for each storage I/O adapter that Extended Adaptive Cache is to be activated on (CCIN 6731 for systems running V4R4 or later).

Because Extended Adaptive Cache is automatically enabled through the RCD, there is no controlled on or off switch. The RCD may be added without system interruption through concurrent maintenance. The RCD resides in an internal disk slot and works with all other disk types and capacities. Be aware that all data in the Extended Adaptive Cache is also guaranteed to be on the disks. In the unlikely event of an RCD failure, there will be no data loss.

The Read Cache Device can be purchased wherever iSeries hardware is sold, or contact your local IBM representative.

Workload Estimator for iSeries

The Workload Estimator

helps you size system needs based on estimated workloads for specific workload types. PM/400 is an integrated OS/400 function that users under processor warranty or on an IBM maintenance agreement can activate for no additional charge. In return, you receive capacity and performance analysis graphs useful in planning for and managing system growth and performance.

The Workload Estimator and PM/400 have been enhanced to work with one another. Through a web-based application, you can size the upgrade to the required iSeries system that accommodates your existing system's utilization, performance, and growth as reported by PM/400. As an additional option, sizings can also include capacity for adding specific applications like Domino, Java, and WebSphere, or the consolidation of multiple AS/400 or iSeries traditional OS/400 workloads on one system. This capability allows you to plan for future system requirements based on existing utilization data coming from your own system.

iSeries Navigator for Wireless

iSeries Navigator for Wireless lets you remotely monitor system performance and status using an Internet-ready telephone, a personal digital assistant (PDA) with a wireless modem, or a traditional Web browser. With your wireless device, you can:

- Run commands across multiple systems
- Start and view system, job, and message monitors
- Work with jobs and messages from the monitors (hold, release, end, reply, get details)
- Manage integrated xSeries servers

For an overview of how iSeries Navigator for Wireless can help you get started with remote monitoring, see the topic iSeries Navigator for Wireless.

For complete and up-to-date information about remote monitoring, see the iSeries Navigator for Wireless

home page.

PATROL for iSeries (AS/400) - Predict

The PATROL for iSeries (AS/400) - Predict product can help you manage iSeries performance by automating many performance management administration tasks. This product offers an in-depth view of current and historical system data for groups of iSeries servers, allowing you to drill down to specific details like CPU and file utilization and status. Additionally, Patrol offers several options for automation that will allow it to proactively address performance and availability problems on your iSeries server before they occur.

This product can help you plan future upgrades and manage the growth of your iSeries environment by offering detailed capacity planning information.

For more information, refer to the BMC products web site.

Scenarios: Performance

One of the best ways to learn about performance management is to see examples illustrating how many of the applications and functions can be used in a sample business environment. Use the following scenarios and configuration examples to learn more about managing performance.

Scenario: Improve system performance after an upgrade or migration
In this scenario, you have just upgraded or migrated your system and it now appears to be running slower than before. This scenario helps you identify and fix your performance problem.

Scenario: System monitor
See an example system monitor that alerts you if the CPU utilization gets too high and temporarily holds any lower priority jobs until more resources become available.

Scenario: Message monitor
See an example message monitor that displays any inquiry messages for your message queue that occur on any of your iSeries servers. The monitor opens and displays the message as soon as it is detected.

Scenario: Job monitor
See an example job monitor that tracks the CPU utilization of a specified job and alerts the job's owner if CPU utilization gets too high.

Related information

Listed below are the iSeries manuals (sometimes called "white books") and IBM Redbooks$^{(TM)}$, in PDF format, that relate to the Performance topic. You can also view or print any of the following PDFs:

- **Manuals**

  Performance Tools for iSeries

  This book provides the programmer with the information needed to collect data about the system, job, or program performance. It also includes tips for printing and analyzing performance data to identify and correct inefficiencies that might exist as well as information about the Manager and Agent features.
- **Web sites**

  iSeries Performance Capabilities Reference

  This reference provides highly technical information about server performance useful for performance benchmarking, capacity planning, and planning for server performance.
- **Redbooks:**
  - **IBM eserver iSeries Universal Connection for Electronic Support and Services**

    This document introduces Universal Connection. It also explains how to use the variety of support tools that report inventories of software and hardware on your machine to IBM so you can get personalized electronic support, based on your system data.

- **Lotus Domino for AS/400: Performance, Tuning, and Capacity Planning**

  This document describes a methodology for performance management. It includes setting up performance objectives, collecting and reviewing performance data, tuning of resources, and capacity planning. Performance guidelines and application design tips are also provided.
- **AS/400 Performance Management**

  This document describes a methodology for performance management. It includes setting up performance objectives, collecting and reviewing performance data, tuning of resources, and capacity planning. Performance guidelines and application design tips are also provided.
- **AS/400 HTTP Server Performance and Capacity Planning**

  The Internet and Web browser-based applications have had a profound effect on how organizations distribute information, perform business processes, service customers, and reach new markets. This book is intended for iSeries programmers, network and system management professionals, and other information technologists who are responsible for designing, developing, and deploying Web-based applications and information systems.
- **Java and WebSphere Performance on IBM eserver iSeries Servers**

  This document provides tips, techniques, and methodologies for working with Java and WebSphere Application Server performance-related issues with a specific focus on iSeries servers.
- **Management Central: A Smart Way to Manage AS/400 Systems**

  Discover the benefits of Management Central. This book discusses Collection Services, the replacement to the performance monitor. It also explains how Management Central allows operators and administrators to monitor servers in a network by providing real-time performance-monitoring capabilities, such as notification of events and automatic responses to events.
- **Managing AS/400 V4R4 with Operations Navigator**

  This book gives you insight into the wide range of iSeries functions that are available through Operations Navigator, now known as iSeries Navigator. In addition to the full discussion of these functions, this book specifically discusses monitoring system performance with Management Central and Collection Services.
- **AS/400 Performance Explorer Tips and Techniques**

  This document provides descriptions and detailed examples of the performance explorer capabilities that were available for V3R6. Specific application examples and reports are provided.

For complete information about iSeries performance, be sure to see the Performance topic.

Printed in U.S.A.