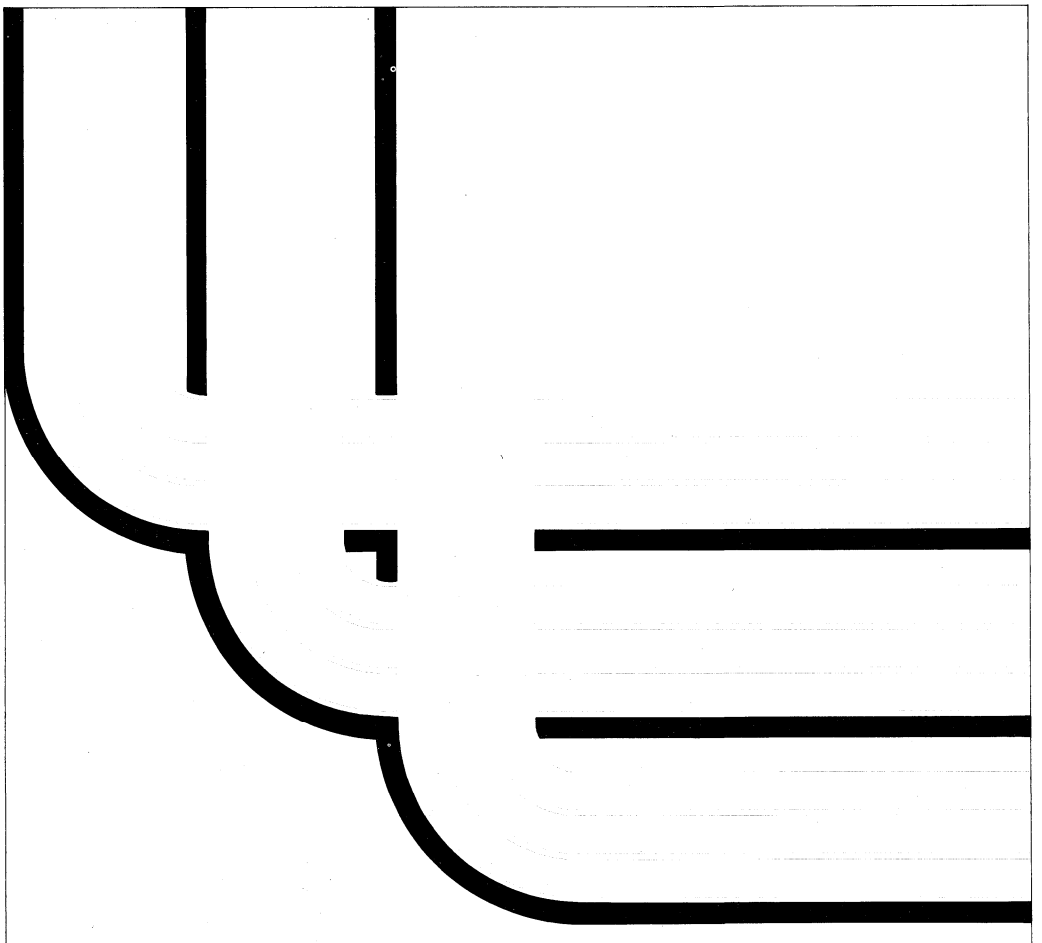
Application System/400

SC41-8078-02

Programming: Work Management Guide

Version 2



System and Application Support

Application System/400

SC41-8078-02

Programming: Work Management Guide

Version 2

Take Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page vii.

Third Edition (November 1993)

This edition applies to the licensed program IBM Operating System/400, (Program 5738-SS1), Version 2 Release 3 Modification 0, and to all subsequent releases and modifications until otherwise indicated in new editions. This major revision makes obsolete SC41-8078-01. Make sure you are using the proper edition for the level of the product.

Order publications through your IBM representative or the IBM branch serving your locality. Publications are not stocked at the address given below.

A Customer Satisfaction Feedback form for readers' comments is provided at the back of this publication. If the form has been removed, you can mail your comments to:

Attn Department 245
IBM Corporation
3605 Highway 52 N
Rochester, MN 55901-7899 USA

or you can fax your comments to:

United States and Canada: 800+937-3430
Other countries: (+1)+507+253-5192

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you or restricting your use of it.

Contents

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Commercial Relations, IBM Corporation, Purchase, NY 10577, U.S.A.

This publication could contain technical inaccuracies or typographical errors.

This publication may refer to products that are announced but not currently available in your country. This publication may also refer to products that have not been announced in your country. IBM makes no commitment to make available any unannounced products referred to herein. The final decision to announce any product is based on IBM's business and technical judgment.

Changes or additions to the text are indicated by a vertical line (|) to the left of the change or addition.

Refer to the "Summary of Changes" on page xi for a summary of changes made to the Operating System/400 operating system and how they are described in this publication.

This publication contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

This publication contains small programs that are furnished by IBM as simple examples to provide an illustration. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. All programs contained herein are provided to you "AS IS". THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE EXPRESSLY DISCLAIMED.

If you are viewing this manual from a compact disk (CD-ROM), the photographs and color illustrations do not appear.

Programming Interface Information

This guide is intended to help the customer manage work on the AS/400 system. This guide primarily documents General-Use Programming Interface and Associated Guidance Information provided by the Operating System/400 program.

General-Use programming interfaces allow the customer to write programs that obtain the services of the OS/400 program.

However, this guide also documents Product-Sensitive Programming Interface and Associated Guidance Information.

Product-Sensitive programming interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of this IBM software product. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-Sensitive programming interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service.

Trademarks and Service Marks

The following terms, denoted by an asterisk (*) in this publication, are trademarks of the IBM Corporation in the United States or other countries or both:

| Advanced Program-to-Program Networking
Application System/400
| APPN
AS/400
| DRDA
IBM
| ILE
| Information Assistant
| Integrated Language Environment
OfficeVision
OfficeVision/400
Operating System/400
Operational Assistant
OS/400
RPG/400
SAA
Systems Application Architecture
SystemView
400

The following terms, denoted by a double asterisk (**) in this publication, are trademarks of other companies as follows:

ROLM	Siemens
BEST/1	BGS Systems, Inc.

About This Guide

This guide helps you set up your initial work management environment and change work management objects to meet your needs. This guide also discusses collecting performance data, tuning your performance, and gathering data on system users and resources.

This guide is organized to provide the simplest possible approach to AS/400 work management. The guide begins with a high-level overview of work management concepts and objects, followed by chapters that detail work management components. In most chapters a "How To's" section provides approaches to various tasks you may want to perform on the system. Throughout the guide are examples of many of these tasks.

You may need to refer to other IBM manuals for more specific information about a particular topic. The *Publications Guide*, GC41-9678, provides information on all the manuals in the AS/400 library.

Who Should Use This Guide

This guide is intended for the programmer or application programmer.

Before using this guide, you should be familiar with general programming concepts and terminology, and have a general understanding of the AS/400 system and the OS/400 licensed program. You should also be familiar with the display stations and printers you are using.

Summary of Changes

This guide is updated to include information on the enhanced function of the AS/400 operating system. The following is a brief summary of those changes:

Security Value 50: A new value of '50' is added to the QSECURITY system value. In addition to the security and integrity offered at level 40, full parameter validation is in effect.

Duplicate Jobs: Have you ever done a `wrkjob`, only to find that multiple jobs share the same simple name? Previous to Version 2 Release 3 you would receive a series of messages listing all the jobs that match. Now if there are other jobs on the system that match the portion of the job name you entered, the Select Job display appears. This display allows you to select the job you want from a list of duplicate job names.

Integrated Language Environment* (ILE*) Changes: The Integrated Language Environment (ILE) compiler constructs bind modules as well as independent programs for all high-level languages that conform with the ILE program. For that reason, the Display Job display now displays programs and procedures, whether they run independently or not. The Display Program Stack display is now the Display Call Stack display. The Display Call Stack Detail display provides information about the activation group involved, the name of the program or procedure, and a statement number.

System Tuning by Storage Pool Paging: In addition to using storage pool sizes and activity levels for performance tuning, you now can tune the system using the paging attribute on the Change Shared Pool (CHGSHRPOOL) command or on the Change Pool Attributes (QUSCHGPA) API. A new API, Change Shared Pool (QWCCHGTN), is also available for tuning by storage pool paging.

New System Values:

- **QALWUSRDMN** contains the names of libraries which allow *USRSPC, *USRIDX, and *USRQ objects.
- **QAUDCTL** contains the on/off switch for auditing.
- **QAUDENDACN** contains the action for the system when audit records cannot be sent to the auditing journal due to errors that occurred when the journal entry was sent.
- **QAUDFRCLVL** controls the number of journal entries written to the security auditing journal before data is forced into auxiliary storage.
- **QCRTOBJAUD** contains the default auditing value for newly created objects.
- **QJOBMSGQFL** defines the action to be taken when a message queue is full.
- **QJOBMSGQMX** specifies the maximum size of a job message queue. When this maximum size is reached for any job message queue, that job message queue is considered full and action must be taken.
- **QSRTSEQ** contains the default sort sequence to be used by the system.

New Performance Data Files:

- **QAPMDDI**: Distributed data interface (DDI) counter
- **QAPMFRLY**: Frame relay counter
- **QAPMSTND**: DDI station counter
- **QAPMSTNY**: Frame relay station counter

Work Management APIs: See Appendix D, "Work Management APIs" on page D-1 for a list of current work management application programming interfaces (APIs).

Chapter 1. Introduction

This chapter describes subsystems and objects in simple terms and shows how they relate to each other. This chapter is a high-level overview intended for users unfamiliar with the concepts of work management.

Work management supports the commands and internal functions necessary to control system operation and the daily workload on the system. In addition, work management contains the functions you need to distribute resources for your applications so that your system can handle your applications.

This guide is designed to show you how the IBM* AS/400* system manages jobs and allocates resources to those jobs.

Why Learn about Work Management

Because IBM ships all AS/400 systems with everything necessary to run typical operations, you are not required to learn about work management to use your system. At some time, however, you may want to change the way your system manages work to better meet your needs, affect the order your jobs are run, solve a problem, improve the system's performance, or simply look at jobs on the system. To make such changes and meet your processing requirements, you need to know which objects affect which pieces of the system and how to change those objects.

Work Management Overview

Work management functions control all the work done on the system. When the Operating System/400* (OS/400*) licensed program is installed, it includes a work management environment that supports all interactive and batch work.

The OS/400 licensed program allows you to tailor this support or to create your own work management environment. To do this, you need an understanding of the work management concepts.

A Simple System

Figure 1-1 shows the concept of a simple system. The purpose of the system is to perform work. Work enters, work is processed, and work leaves the system. If you think of work management in these three terms, work management will be easier to understand. **Work management** describes where work enters the system, where and with what resources work is processed, and where output from work goes.

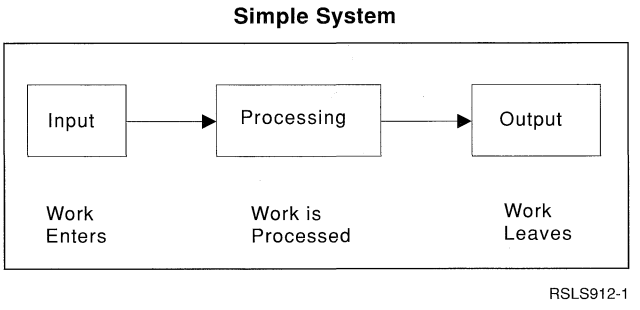


Figure 1-1. Simple System

Subsystems

A **subsystem** is an environment where work and resources are managed to get work done. An AS/400 system can contain several subsystems, all operating independently of each other, but sharing system resources.

Each subsystem can run unique operations. For instance, you can set up one subsystem to handle only interactive jobs, while another subsystem handles only batch jobs. Subsystems can also be designed to handle many types of work. The system allows you to decide the number of subsystems and what types of work each subsystem will handle.

Subsystem Descriptions

A **subsystem description** is a system object that contains information defining the characteristics of an operating environment controlled by the system. Like a set of detailed blueprints, each subsystem description is unique, containing the specific characteristics describing the subsystem. The description includes where work can enter the

subsystem, how much work the subsystem can handle, how much main storage (memory) will be used, and how quickly jobs in the subsystem can run. You can use a subsystem description supplied with your system (with or without making changes to it), or you can create your own. See "How To's" on page 3-16 for information about creating and changing subsystem descriptions.

Work Entries

The subsystem description describes the **work entry**, the location where work can enter the subsystem. Work entries are classified as follows:

Autostart job entry
Identifies the jobs to start as soon as the subsystem starts.

Communications entry
Identifies the communications device that another system uses to submit work.

Job queue entry
Identifies the job queue from which to take work and determines how much work to accept.

Prestart job entry
Reduces the amount of time required to perform a program start request received through a communications entry. A prestart job starts before a program start request is sent by another system, thus decreasing the initial processing time for a communications job.

Work station entry
Identifies the work station from which to take work.

Pools

Pools contain main storage. You can control the number and size of the pools in order to control how much work can be done in a subsystem. The greater the size of the pools in a subsystem, the more work can be done in the subsystem.

The two types of pools in a system are shared pools and private pools. Shared pools are either special or general; the machine pool and base pool are considered special shared pools, all other shared pools are considered general.

Shared Pools Special shared pools include the following:

- The machine pool provides storage for jobs the system must run that do not require your attention. The machine pool provides a separate space for high-use system jobs and objects.
- The base pool contains storage that can be shared by many subsystems. The base pool gets all the leftover main storage not used by any of the other pools.

General shared pools are pools of main storage that multiple subsystems can use at the same time.

Private Pools Private pools are pools of main storage that cannot be shared by multiple subsystems. Often simply referred to as a pool, a private pool contains a specified amount of storage to be used by only one subsystem. You can have as many as 14 private pools allocated for use in active subsystems.

Jobs

Each piece of work in a system is called a **job**, and each job has a unique name. A job can enter the subsystem from any of the work entries (a job queue entry, work station entry, communications entry, autostart job entry, or prestart job entry). All user jobs run within subsystems.

There are two basic types of jobs: interactive jobs and batch jobs. An **interactive job** starts when you sign on to the system from a display station and ends when you sign off. Working from a display station, you interact with the system by issuing commands, using function keys, and running programs and applications. A **batch job** on the other hand, needs little or no interaction from you in order to run. It can enter a subsystem from a job queue, communications entry, autostart job entry, or a prestart job entry. For example, you can submit a job to run as a batch job while you continue to work from a display station.

Job Descriptions

You can create **job descriptions** to describe batch jobs or interactive jobs. You can create unique descriptions for each user of the system.

Job descriptions contain job attributes such as (the following is a partial list):

- User ID
- Job queue (JOBQ)
- JOBQ priority
- Job date
- Job date format
- Hold on JOBQ
- Print device
- Output queue
- Output priority
- Routing data
- Accounting code

Routing Entries and Routing Data

Routing entries and routing data provide information on starting a job in a subsystem. The **routing entry** identifies the main storage subsystem pool to use, the controlling program to run (typically the system-supplied program QCMD), and additional run-time information (stored in the class object). Routing entries are stored in the subsystem description. **Routing data** identifies the routing entry the job uses. For most jobs started on the system, default routing data is used to allow typical functions to occur. However, your work management is more efficient if you control the routing data value so that the job has the proper run attributes. Routing data is stored in the job description.

Classes

Classes allow you to tell the system which jobs are most important. Classes provide information about how jobs are handled while in a subsystem. They contain information such as the priority of a job while it's running. By specifying different classes for different job types, you can establish some guidelines under which a subsystem performs.

Using classes, you can also control how long certain jobs can run in a subsystem and how much storage those jobs can use.

System Values

System values are pieces of information that affect the operating environment in the entire system. The AS/400 system has system values for the following kinds of information:

- Date-and-time values
- Editing values
- System control values
- Library list values
- Allocation values
- Message and logging values
- Storage system values
- Security system values

User Profiles

A **user profile** holds information that is specific to one user. The AS/400 user profiles contain two kinds of information: authority information and job attributes.

Authority information allows you to assign specific authority levels to each system user. You can tell the system to only allow users with particular authority levels access to certain AS/400 objects. The system looks in the user profile to determine whether to grant the user access to the objects they requested.

Job attributes allow you to tailor certain job attributes to specific users. For example you might want all output for a particular user to go to one output queue or you may want one user to use a special job description. The following are a few of the job attributes found in a user profile:

- Job description
- Output queue or printer device
- Accounting code
- Maximum scheduling priority

Overview of Starting a Subsystem

To start the subsystem, use the Start Subsystem (STRSBS) command, which specifies the name of the subsystem description to use. When the command is issued, the system allocates the following items:

1. Pools of main storage
2. Display stations
3. Communications devices

4. Job queues

Next, the autostart jobs and prestart jobs are started and the subsystem is ready for work. Figure 1-2 shows what occurs when a subsystem starts.

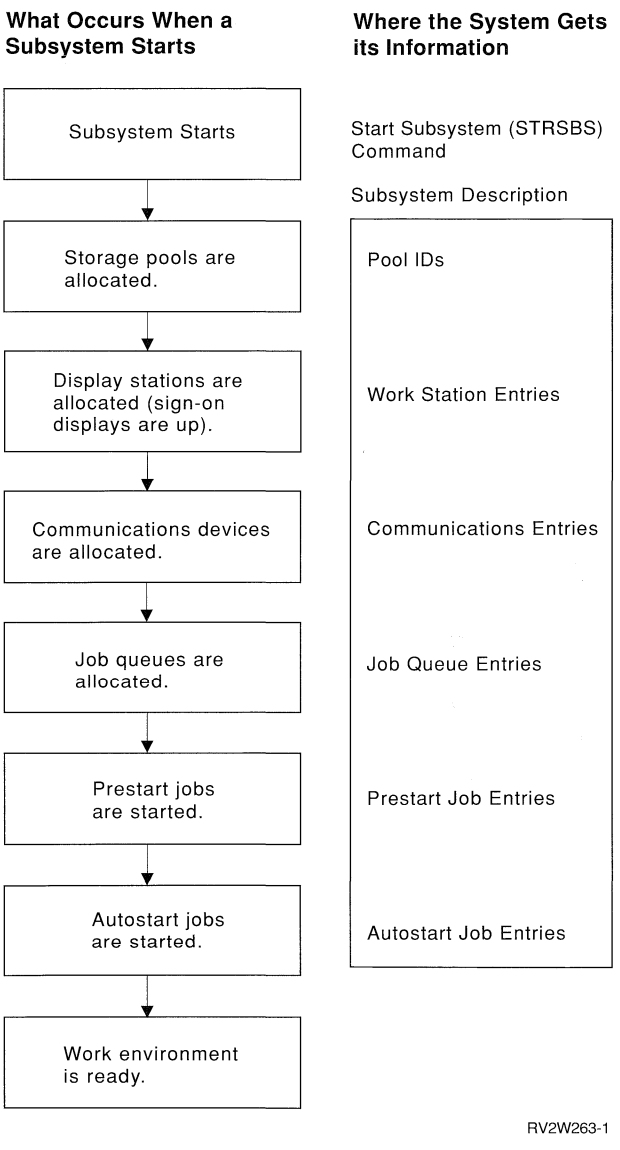


Figure 1-2. Starting a Subsystem

The system relies on subsystem descriptions when starting subsystems. Therefore, if you want to change the amount of work (number of jobs) coming from a job queue, for example, you only need to change the job queue entry in the subsystem description. For a description of how to create and change subsystem descriptions, see "Subsystem Descriptions" on page 3-1.

Overview of Starting an Interactive Job

When an interactive job is started by a user signing on to the system, the subsystem first looks in the work station entry for the job description in order to get the attributes for the interactive job. If the work station entry specifies *USRPRF for the job description, the job will use the information from the user profile. This flexibility allows you to specify whether the job's attributes are tied to the work station or to the individual user.

Once the subsystem knows which job description to use, it may not find all the job attributes in the job description. Some attributes may be in the user profile. The user profile contains job attributes that allow you to tailor certain things specifically for that user. If the user profile does not have the information, the subsystem looks at the system value.

After the subsystem gathers all of the job's attributes, it checks the job description for the routing data. The subsystem uses the routing data to find a routing entry in the subsystem description. The routing entry provides information about which pool the job will use, which routing program will be used, and from which class the job will get its runtime attributes.

When all of these pieces are obtained, the routing program runs. IBM supplies a routing program called QCMD, which you can use for all types of work. QCMD knows if the job is an interactive job and checks the user profile for an initial program to run. If the initial program finishes running, QCMD displays the initial menu. Figure 1-3 on page 1-5 shows what occurs when an interactive job starts.

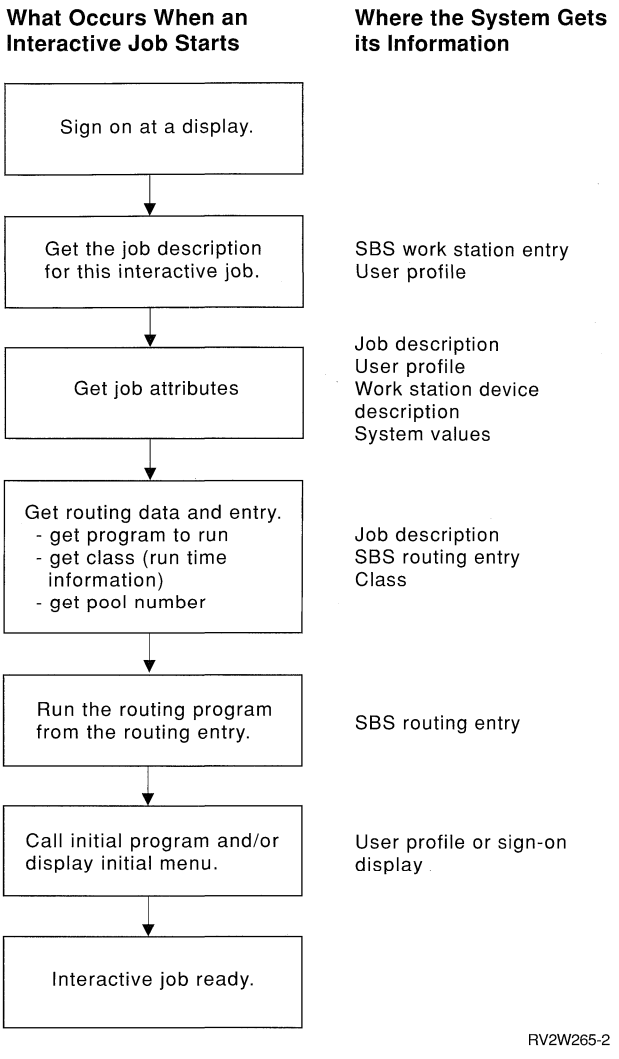


Figure 1-3. Starting an Interactive Job

Overview of Starting Batch Jobs

You can start a batch job when you submit a job using the submit job (SBMJOB) command, which has parameters that allow you to specify some of the job attributes.

If the job attributes are not found on the submit job command, the job will look in the job description (specified on the SBMJOB command), the current user's user profile, and the currently active job (the job issuing the SBMJOB command). Similar to interactive job initiation, you can specify in the job description to use the user profile, and the user profile can specify to use a system value to find certain job attributes. Figure 1-4 shows what occurs when a batch job starts.

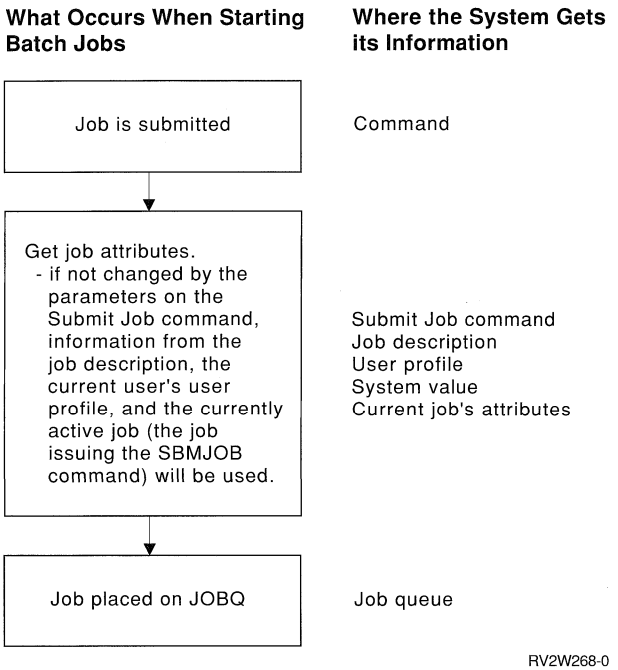


Figure 1-4. Starting Batch Jobs

Once the job has all its attributes, it resides on the job queue.

When the subsystem is ready to handle a job, it looks for jobs in the job queues (those that the subsystem has allocated). Then, like interactive job processing, the subsystem checks the job description for the routing data. The subsystem uses the routing data to find a routing entry. The routing entry provides information about which pool the job will use, which routing program will be used, and in which class the job will run.

After this information is obtained, the routing program is run. If you use QCMD, QCMD will carry out the SBMJOB command. Figure 1-5 on page 1-6 shows the steps of the SBMJOB command.

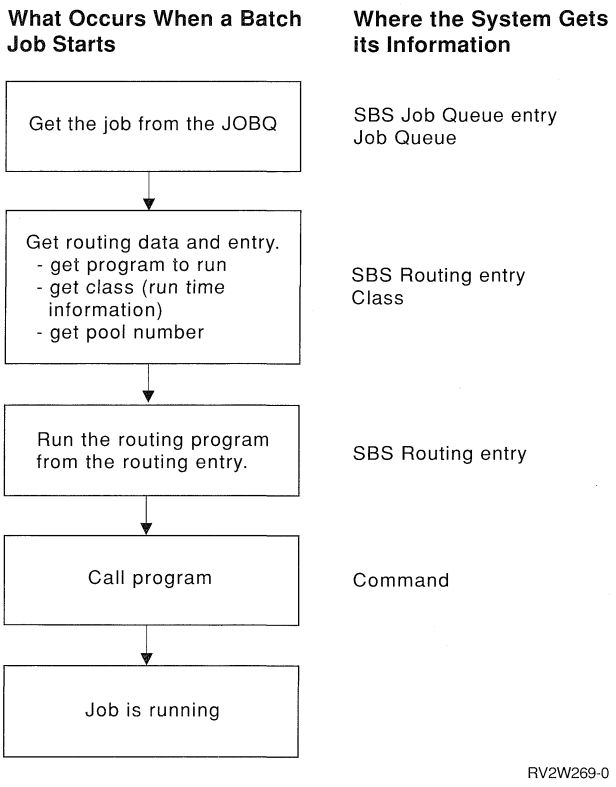


Figure 1-5. When a Batch Job Starts

Controlling Job Attributes

The following information tells you how the subsystem knows when to get job attributes from a job description, user profile, system value, or the currently active job.

You can specify attributes from the job description, user profile, or system value by using the following information.

***JOBD** Tells the job to get its attributes from the job description.

***USRPRF** Tells the job to get its attributes from the user's user profile.

***SYSVAL** Tells the job to get its attributes from a system value.

***CURRENT** Tells the job to get its attributes from the job issuing the submit job (SBMJOB) command.

***WRKSTN** Tells the job to get its attributes from the workstation with the job (interactive jobs only).

You can take advantage of this flexibility. This allows you to control jobs at the job level, user level, or system level. You may have your system set up to go all the way to the system value to get job attributes (which is how the system defaults and is sent to you). Then, if you want to change a value for everyone on the system all you need to do is change the system value and everyone's jobs are affected. If you specify *USRPRF, you can control or affect only the users you want to change. By specifying a value in a job description you can affect all the types of jobs that use that job description. For example, all batch jobs use the same job description. Changing the job description for the batch jobs would affect all of your batch jobs and leave all other jobs unaffected.

Note: Be aware that specifying *USRPRF and *SBSD can sometimes mean that the system should use the name of the object not that the system should get job attributes from those objects.

Summary of Work Management Objects and System Values

AS/400 work management allows you to control the way work is managed on your system. Because of the large amount and complexity of function on the system, you should learn about work management in stages. You should begin by deciding what work you need the system to do and how you want that work done.

Because the AS/400 system provides you with default values for all parameters, you can start using your system to perform work right away, without changing any defaults. As you learn more about ways work management can help you optimize the system, you can begin changing the defaults to values that work best for your unique processing environment.

By keeping it simple from the start and learning work management in stages, you can avoid becoming frustrated by the flexibility in the system. As you become comfortable with the concept of work management, and begin to understand how each of the pieces introduced in this chapter work, you will be on your way to getting the most from your AS/400 system.

Examples and Command Summary

Figure 1-6 on page 1-7 shows examples of work management objects and system values and their related commands. Use this table as a quick reference.

Figure 1-6. Examples and Command Summary

Objects and Values	Example	Command
System Values	QSECURITY	WRKSYSVAL
Subsystem Descriptions	QSYS/QBASE QSYS/QINTER QSYS/QBATCH	WRKSBSD
Job queues	QGPL/QBATCH	WRKJOBQ
Job descriptions	QGPL/QDFTJOBD QGPL/QBATCH	WRKJOBD
Classes	QGPL/QINTER QGPL/QBATCH	WRKCLS
Jobs	010457/QSECOFR/DSP01	WRKACTJOB WRKUSRJOB WRKSBSJOB SBMJOB WRKSBMJOB WRKJOB
User profiles	QSECOFR QSYSOPR	WRKUSRPRF
Output queues	QGPL/QPRINT	WRKOUTQ

Chapter 2. System Values and Network Attributes

This chapter provides a summary of all AS/400 system values and network attributes as well as a detailed description of each system value. This chapter also describes how to use system values to control or change the overall operation of the system.

System values contain specifications that can be used to control or change the overall operation of your system. System values are not objects and cannot be passed as parameter values like CL variables.

A system value can be placed in a CL variable for use in a CL program using the Retrieve System Value (RTVSYSVAL) command. (The *CL Programmer's Guide* contains more information on using CL variables in programs.)

This chapter provides you with a summary of system values arranged by the types, or categories, that appear on the Work with System Values display:

- Date-and-Time
- Editing
- System Control
- Library List
- Allocation
- Message-and-Logging
- Storage
- Security

A summary table for each category lists each system value with a page reference to the detailed description. The detailed descriptions appear later in the chapter.

Also contained in this chapter is a list of network attributes shipped with the system. "How To's" on page 2-44 provides you with directions for displaying, changing, and retrieving system values and network attributes.

You can also save system values using the Save System (SAVSYS) command. Refer to the *Basic Backup and Recovery Guide* for information about saving system values.

Short Descriptions of System Values

The following is a summary of all the system values and their initial values shipped with the system.

Date-and-Time System Values

The date-and-time system values allow you to control the date and time on your system. Figure 2-1 shows these values.

Figure 2-1 (Page 1 of 2). Date-and-Time System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QDATE		System date	2-11	Character	5[1] or 6
QYEAR		Year	2-11	Character	2
QMONTH		Month of the year	2-11	Character	2
QDAY		Day of the month (or year[1])	2-12	Character	2 or 3[1]
QLEAPADJ	'0'	Leap year adjustment	2-12	Decimal	(5 0)
QTIME		Time of day	2-13	Character	6, 7, 8, or 9[2]
QHOUR		Hour of the day	2-13	Character	2
QMINUTE		Minute of the hour	2-13	Character	2
QSECOND		Second of the minute	2-13	Character	2

Figure 2-1 (Page 2 of 2). Date-and-Time System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QUTCOFFSET	'+0000'	Coordinated universal time offset. The number of hours and minutes current system time is offset from coordinated universal time.	2-12	Character	5

1 For Julian dates
2 For seconds, tenths, hundredths, and thousandths of a second

Editing System Values

The editing system values allow you to control the format for displaying currency symbols, dates, and decimal values on your system. Figure 2-2 shows these values.

Figure 2-2. Editing System Values

Name	Shipped Value[1]	Description	Reference Page	Type	Length
QCURSYM	'$'	Currency symbol	2-14	Character	1
QDATFMT	'*MDY*'	Date format	2-14	Character	3
QDATSEP	'/'	Date separator	2-14	Character	1
QDECFMT	' '	Decimal format	2-14	Character	1
QTIMSEP	':'	Time separator	2-15	Character	1

2-15

1 The shipped value may be different in different countries.

System Control System Values

The system control system values allow you to control or display information specific to your system. Figure 2-3 shows these values.

Figure 2-3 (Page 1 of 5). System Control System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QABNORMSW	'0'	Previous end of system indicator. ('0' means previous end was normal. '1' means previous end was abnormal.) Cannot be changed.	2-15	Character	1
QASTLVL	'*BASIC'	Assistance level. Specifies the level of system displays available.	2-15	Character	10

Figure 2-3 (Page 2 of 5). System Control System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QATNPGM	'*ASSIST'	Attention program. ('*ASSIST' means that the Operational Assistant* menu will appear. '*NONE' means Operational Assistant menu will not appear. 'pgm lib' means the program and library that will be called when the Attention key is pressed.	2-15	Character	20
QAUTOCFG	'1'	Automatic configuration indicator. ('0' means automatic configuration is off. '1' means automatic configuration is on.)	2-16	Character	1
QAUTOVRT	0	Number of virtual devices to be automatically configured.	2-16	Decimal	(5 0)
QCCSID	'65535'	Coded character set identifier.	2-16	Decimal	(10 0)
QCHRID	'697 037'[1]	Default graphic character set and code page used for displaying or printing data.	2-17	Character	20
QCMNRCYLMT	'0 0'	Provides recovery limits for system communications recovery.	2-17	Character	20
QCNTRYID	US[1]	Country identifier.	2-18	Character	2
QCONSOLE	'QCONSOLE'	Console name. Cannot be changed.	2-18	Character	10
QCTLSBSD	'QBASE QSYS'	Controlling subsystem name.	2-18	Character	20
QDBRCVYWT	'0'	Database recovery indicator. ('0' means do not wait. '1' means wait.)	2-18	Character	1
QDEVNAMING	'*NORMAL'	Indicates the device naming convention. ('*NORMAL' means follow AS/400 standards. '*S36' means follow System/36 standards. '*DEVADR' means derive device name from the device address.)	2-19	Character	10
QDEVRCYACN	'*MSG'	Specifies what action to take when an I/O error occurs for the job's requesting program device.	2-19	Character	20
QDSCJOBITV	'240'	Time interval, in minutes, that a job can be disconnected before it ends.	2-19	Character	10

Figure 2-3 (Page 3 of 5). System Control System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QIGC		Indicates whether the DBCS version of the system is installed. ('1' means the DBCS version is installed. '0' means the DBCS version is not installed.) Cannot be changed.	2-20	Character	1
QIGCCDEFNT	'*NONE'	Specifies the value of the DBCS font name that should be used when transforming SCS DBCS data into advanced function printing data stream (AFPDS) data and when creating an AFPDS spooled file with shift in/shift out (SI/SO) characters present in the data.	2-20	Character	20
QIPLDATTIM	'*NONE'	Date and time to automatically IPL the system.	2-20	Character	20
QIPLSTS	'0'	IPL status indicator. ('0' means the last IPL was an operator panel IPL. '1' means it was an auto-IPL after power was restored. '2' means it was the result of PWRDWNSYS RESTART(*YES). '3' means it was an auto-IPL at a specified time of day. '4' means it was a remote IPL.) Cannot be changed.	2-20	Character	1
QIPLTYPE	'0'	Indicates type of IPL to perform. ('0' means unattended IPL. '1' means attended IPL with service displays. '2' is also valid but should only be used if directed by IBM.	2-21	Character	1
QKBDBUF	'*TYPEAHEAD'	Specifies the keyboard buffering value to be used when a job is initialized. (*TYPEAHEAD means the type-ahead feature is on and the Attention key is off, *NO means both the type-ahead feature and Attention key are off, *YES means the type-ahead feature and Attention key are on.)	2-21	Character	10
QKBDTYPE	'USB'[1, 2]	Specifies a language character set for the keyboard.	2-21	Character	3
QLANGID	'ENU'[1]	Language identifier.	2-22	Character	3

Figure 2-3 (Page 4 of 5). System Control System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QMODEL		System model number. Cannot be changed.	2-22	Character	4
QPFRADJ	'2'	Performance adjustment. Specifies whether the system should adjust values during IPL and at regular intervals for system pool sizes and activity levels. ('0' means no performance adjustment. '1' means performance adjustment at IPL. '2' means performance adjustment at IPL and dynamically. '3' means dynamic performance adjustment.)	2-22	Character	1
QPRTDEV	'PRT01'	Default printer device description.	2-23	Character	10
QPRTKEYFMT	'*PRTHDR'	Print key format. Specifies whether border and header information is provided when the print key is pressed. ('*NONE' means border and header information is not provided. '*PRTBDR' means border information is provided. '*PRTHDR' means header information is provided. '*PRTALL' means border and header information is provided.)	2-23	Character	10
QPWRDWNLMT	600	Maximum amount of time (in seconds) allowed for PWRDWNSYS *IMMED.	2-23	Decimal	(5 0)
QPWRRSTIPL	'0'	Auto-IPL after power restored allowed. ('0' means no auto-IPL after power restored. '1' means auto-IPL after power restored.)	2-24	Character	1
QRMTIPL	'0'	Remote power on and IPL indicator. ('0' means remote power on and IPL is not allowed. '1' means remote power on and IPL is allowed.)	2-24	Character	1
QSCPFCONS	'1'	IPL console indicator. ('1' means to switch to unattended IPL if console problems occur during IPL. '0' means end system.)	2-24	Character	1

Figure 2-3 (Page 5 of 5). System Control System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QSPCENV	'*NONE'	Indicates default special environment. ('*NONE' means no special environment. '*S36' means System/36 environment.)	2-24	Character	10
QSRLNBR		System serial number. Cannot be changed.	2-24	Character	8
QSRTSEQ	'*HEX'	Default sort sequence to be used by the system.	2-24	Character	20
QSTRPRTWTR	'1'	Indicates if printer writers should be started. ('0' means do not start printer writers. '1' means start printer writers.) Cannot be changed.	2-25	Character	1
QSTRUPPGM	'QSTRUP QSYS'	Startup program name called from autostart job in the controlling subsystem.	2-25	Character	20
QUPSDLYTIM	'*CALC'	Uninterruptible power supply delay time.	2-25	Character	20
QUPSMSGQ	'QSYSOPR QSYS'	Message queue for uninterruptible power supply messages.	2-26	Character	20

1 The shipped value may be different in different countries.
2 For United States and Canada (see "QKBDTYPE" on page 2-21 for other countries).

Library List System Values

The library list system values allow you to control or display the system and user parts of the library list. Figure 2-4 shows these values.

Figure 2-4. Library List System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QSYSLIBL	'QSYS QSYS2 QHLPSYS QUSRSYS'	System part of the library list.	2-27	Character	150
QUSRLIBL	'QGPL QTEMP'	User part of the library list.	2-27	Character	250

Allocation System Values

The allocation system values allow you to control the number of jobs and storage sizes on your system. Figure 2-5 shows these values.

Figure 2-5. Allocation System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QACTJOB	20	Initial number of active jobs for which to allocate storage.	2-28	Decimal	(5 0)
QADLACTJ	10	Additional number of active jobs for which to allocate storage.	2-29	Decimal	(5 0)
QADLSPLA	2048	Additional storage for extending spooling control block (bytes).	2-29	Decimal	(5 0)
QADLTOTJ	10	Additional total number of jobs for which to allocate storage.	2-29	Decimal	(5 0)
QJOBMSGQFL	'*NOWRAP'	Action the system takes when the value specified for QJOBMSGQMX is reached.	2-29	Character	10
QJOBMSGQMX	16	The maximum size to which a job message queue can grow (MB).	2-30	Decimal	(5 0)
QJOBMSGQSZ	16	This system value no longer affects the operating system.	2-30	Decimal	(5 0)
QJOBMSGQTL	24	This system value no longer affects the operating system.	2-30	Decimal	(5 0)
QJOBSPLA	1536	Initial size of spooling control block for a job (bytes).	2-28	Decimal	(5 0)
QRCLSPLSTG	'8'	Reclaim spool storage.	2-30	Character	10
QTOTJOB	30	Initial total number of jobs for which to allocate storage.	2-27	Decimal	(5 0)

Message-and-Logging System Values

The message-and-logging system values allow you to control messages and how they are logged on your system. Figure 2-6 shows these values.

Figure 2-6 (Page 1 of 2). Message-and-Logging System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QACGLVL	'*NONE'	Accounting level.	2-30	Character	80
QHSTLOGSIZ	5000	Maximum number of records for each version of the history log.	2-31	Decimal	(5 0)
QPRBFTR	'*NONE'	Problem filter.	2-31	Character	20
QPRBHLDITV	30	The minimum retention period for problems in the problem log.	2-31	Decimal	(5 0)
QPRTTXT	'*BLANK'	Up to 30 characters of text that can be printed at the bottom of the form.	2-31	Character	30
QSFWERRLOG	'*LOG'	Software error log.	2-31	Character	10

Figure 2-6 (Page 2 of 2). Message-and-Logging System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QSRVDMP	'*DMPUSRJOB'	Control for requesting dumps: no jobs, system jobs, user jobs, or all jobs.	2-32	Character	10
QSTSMSG	'*NORMAL'	Allows user to suppress status messages.	2-32	Character	10

Storage System Values

The storage system values allow you to control storage size and activity levels on your system. Figure 2-7 shows these values.

Figure 2-7. Storage System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QBASACTLVL	6	Activity level of base storage pool.	2-33	Decimal	(5 0)
QBASPOOL	500	Minimum size of base storage pool (KB).	2-32	Decimal	(10 0)
QMAXACTLVL	'*NOMAX'	Maximum activity level of the system.	2-33	Decimal	(5 0)
QMCHPOOL	1500	Machine storage pool size (KB).	2-32	Decimal	(10 0)
QTSEPOOL	'*NONE'	Time slice end pool.	2-33	Character	10

Security System Values

The security system values allow you to control security measures on your system. Figure 2-8 shows these values.

Figure 2-8 (Page 1 of 4). Security System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QALWUSRDMN	'*ALL'	Determines which libraries on the system may contain *USRSPC, *USRIDX, and *USRQ objects.	2-34	Character	500
QAUDCTL	'*NONE'	Controls when auditing is active.	2-34	Character	50
QAUDENDACN	'*NOTIFY'	Defines the system's action when the audit records cannot be sent to the journal because of errors.	2-34	Character	10
QAUDFRCLVL	'*SYS'	Controls the number of journal entries written to the security auditing journal before data is forced into auxiliary storage.	2-35	Decimal	(5 0)

Figure 2-8 (Page 2 of 4). Security System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QAUDLVL	'*NONE'	Controls what actions are audited on the system.	2-35	Character	160
QCRTAUT	*CHANGE	Create authority. Specifies the default public authority for create (CRTxxx) commands that can be set system-wide.	2-36	Character	10
QCRTOBJAUD	'*NONE'	Specifies the default auditing value used when objects are created in a library.	2-37	Character	10
QDSPSGNINF	'0'	Controls the display of sign-on information. ('0' means the sign-on information is not displayed. '1' means the sign-on information is displayed.)	2-37	Character	1
QINACTITV	'*NONE'	Inactive job time-out in minutes. ('*NONE' means no time-out interval. '5' through '300' is the range of the time-out interval in minutes.)	2-37	Character	10
QINACTMSGQ	'*ENDJOB'	Message queue to receive job inactive messages. ('*ENDJOB' means the inactive job, secondary jobs and/or group jobs will be ended. *DSCJOB means that when the value specified in QINACTITV is reached, all jobs associated with the work station will be disconnected. The system value can also be a message queue name.)	2-38	Character	20
QLMTDEVSSN	'0'	Controls the ability to limit concurrent device sessions. ('0' means the user can sign on at more than one device; '1' means the user cannot sign on at more than one device.)	2-38	Character	1
QLMTSECOFR	'1'	Controls the ability to limit access to work stations for users with *ALLOBJ or *SERVICE special authority. ('0' means that users with *ALLOBJ or *SERVICE special authority can sign-on any device; '1' means that users with *ALLOBJ or *SERVICE special authority cannot sign-on.)	2-38	Character	1
QMAXSGNACN	'3'	Maximum sign-on attempts action. ('1' means vary off device. '2' means disable user profile. '3' means vary off device and disable user profile.)	2-38	Character	1

Figure 2-8 (Page 3 of 4). Security System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QMAXSIGN	'15'	Maximum number of not valid sign-on attempts allowed.	2-39	Character	6
QPWDEXPITV	'*NOMAX'	Controls the number of days a password is valid.	2-39	Character	6
QPWDLMTAJC	'0'	Limits the use of adjacent numeric digits in a password. ('0' means adjacent digits are allowed; '1' means adjacent digits are not allowed.)	2-39	Character	1
QPWDLMTCHR	'*NONE'	Limits the use of certain characters in a password.	2-40	Character	10
QPWDLMTREP	'0'	Limits the use of repeating characters in a password. ('0' means characters can be repeated; '1' means characters cannot be repeated.)	2-40	Character	1
QPWDMAXLEN	10	Controls the maximum number of characters in a password.	2-40	Decimal	(5 0)
QPWDMINLEN	1	Controls the minimum number of characters in a password.	2-40	Decimal	(5 0)
QPWDPOSDIF	'0'	Controls the position of characters in a new password. This prevents the position of the characters from being the same as in previous passwords. ('0' means characters can be used in the same position; '1' means characters cannot be used in the same position.)	2-41	Character	1
QPWDRQDDGT	'0'	Controls whether a numeric digit is required in a new password. ('0' means a numeric digit is not required; '1' means a numeric digit is required.)	2-41	Character	1
QPWDRQDDIF	'0'	Controls whether a password must be different than the 32 previous passwords. ('0' means a password can be the same as any of the previous passwords except the last one; '1' means a password cannot be the same as any of the previous 32.)	2-41	Character	1
QPWDVLDPGM	'*NONE'	Specifies the name of a password validation program supplied by the installation.	2-41	Character	20

Figure 2-8 (Page 4 of 4). Security System Values

Name	Shipped Value	Description	Reference Page	Type	Length
QRMTSIGN	'*FRCSIGNON'	Remote sign-on control. ('*FRCSIGNON' means normal sign-on required. '*SAMEPRF' means source and target user profile are the same. '*REJECT' means no remote sign-on is allowed. '*VERIFY' means verify that user has access to system. The system value can also be a program name.)	2-42	Character	20
QSECURITY	'10'	Indicates security level.	2-42	Character	2

Detailed Description of System Values

System values are provided by IBM. You cannot create them. The following is a list of system values, grouped by category.

Date-and-Time System Values

The date-and-time system values allow you to control the date and time on your system.

Date-and-time system values must always be enclosed in apostrophes.

If you change a system value during any operation that measures the length of time, a negative value may be set if the end time is less than the start time.

QDATE: QDATE is the system date. QDATE is composed of the following system values: QYEAR, QMONTH, and QDAY. The format of the date is as specified in the system value QDATFMT. The date formats that can be specified are YMD, MDY, DMY, (Y = year, M = month, D = day) or JUL (Julian format). Its value can be set from the IPL options display and is updated when the system value QTIME reaches midnight (000000). A change made to this value may also change QYEAR, QMONTH, and QDAY.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Character	5 (Julian) or 6	No shipped value

QYEAR: QYEAR is the system value that specifies the last two digits for the year. Its value can range from '0' through '99'. When you specify the date in CL commands, you can specify only 2 digits for the year in the date format. The system assigns the first 2 digits for the year (YY) based on the following rules:

- If the last two digits of the year are equal to or greater than 40, the system assigns a value of 19 for the first two digits (19YY).
- If the last two digits are less than 40, the system assigns a value of 20 for the first two digits (20YY).

A change to this system value takes effect immediately. Changing this value also affects the system value QDATE.

Type	Length	Shipped Value
Character	2	No shipped value

QMONTH: QMONTH is the system value for the month of the year. It cannot be displayed or changed if the date format specified in system value QDATFMT is Julian (JUL). Its value can range from '1' through '12'. A change to this system value takes effect immediately. Changing this value also changes the system value QDATE.

Type	Length	Shipped Value
Character	2	No shipped value

QDAY: QDAY is the system value for the day of the month. It must be a valid day of the specified month or year (if the date format is Julian). For Julian dates only, QDAY is a 3-character value ('001' through '366'). A change to this system value takes effect immediately. Changing this system value also changes the system value QDATE.

Type	Length	Shipped Value
Character	2 or 3 (Julian)	No shipped value

QLEAPADJ: QLEAPADJ is the system value for leap year adjustment. It is used to adjust the system calendar algorithm for the leap year in different calendar systems. If your calendar year agrees with what is used in the Gregorian calendar system, then this system value should be zero. If your calendar year differs from the Gregorian, you may need to adjust the system calendar algorithm to account for the leap year of that calendar year you are using. To make the adjustment, divide the leap year in your calendar system by 4; then set QLEAPADJ to the value of the remainder.

Example: The Gregorian calendar year of 1988 was the year 77 in the Republic of China calendar. Because 77 was a leap year for the Republic of China, you need to divide 77 by 4; this leaves a remainder of 1. Therefore, to adjust the system calendar algorithm for the Republic of China, specify a 1 for the QLEAPADJ system value.

Changing the QLEAPADJ system value does not change the system clock and job dates of active jobs, but it may change the QDATE system value.

Type	Length	Shipped Value
Decimal	(5 0)	May be different for different countries

QUTCOFFSET: QUTCOFFSET is the system value indicating difference in hours and minutes between Universal Time Coordinated (UTC), also known as Greenwich mean time (GMT), and the current system (local) time.

This is the number of hours and minutes you need to subtract from local time to obtain the UTC. This value is 5 characters long. The first character is a plus (+) or minus (–) sign. The next 2 characters specify hours ranging from 00 through 24. The last 2 characters specify minutes ranging from 00 through 59 (if it is less than 24 hours).

QUTCOFFSET is used by the system when processing alerts that are sent to other systems in a network. Systems in a network may be set up to signal alerts to one main system within the network when a problem arises. If these systems span across time zones, the time sent with the alert will be different than the time of the main system. To correct this, the QUTCOFFSET value is sent in the alert.

The WRKALR command does not display the offset, but customer applications that parse alerts can make use of the UTC offset, since it has the Systems Network Architecture (SNA) format. This offset can also be used by other customer applications if they need to know how the time on the system differs from another system across a time zone.

Suppose you have multiple systems in a network: the main system in Richmond, Indiana (Eastern time zone); one in Rochester, Minnesota (Central); and one in Los Angeles, California (Pacific). QUTCOFFSET would be set to '-0500' on the Richmond system, '-0600' on the Rochester system, and '-0800' on the Los Angeles system. Each system could use its local time for the system time. You could use QUTCOFFSET to calculate a common time among all the systems.

A change to this value takes effect immediately.

Type	Length	Shipped Value
Character	5	+0000

QTIME: QTIME is the system value for the time of day. QTIME is composed of the following system values: QHOUR, QMINUTE, and QSECOND. Its value can be set from the IPL options display. QTIME has the format hhmmss, where hh = hours, mm = minutes, and ss = seconds. A change made to this value takes effect immediately, and may affect the system values QHOUR, QMINUTE, and QSECOND.

Type	Length	Shipped Value
Character	6, 7, 8, or 9	No shipped value

QTIME is a 6- to 9-character value depending on the resolution required. A length of 6 characters provides 2 characters each for hours, minutes, and seconds. As the length increases, the time resolution becomes more precise (tenths of a second in position 7, hundredths of a second in position 8, and thousandths of a second in position 9).

The Retrieve System Value (RTVSYSVAL) command allows the CL variable to be greater than or equal to 6 characters. If the CL variable is larger than 9 characters, then the first 9 characters contains hours, minutes, seconds, and milliseconds and the others are padded with blanks. The Change System Value (CHGSYSVAL) command sets milliseconds to zero on a change to QTIME, QHOUR, QMINUTE, or QSECOND. Milliseconds cannot be specified on any system value change. The Display System Value (DSPSYSVAL) command does not display milliseconds.

The following table shows the various values provided by the CL character variable.

Length of CL Character Variable	Meaning
<6	Not valid.
6	Hours, minutes, and seconds.
7	Hours, minutes, seconds, and tenths of a second.
8	Hours, minutes, seconds, tenths of a second, and hundredths of a second.
9	Hours, minutes, seconds, tenths of a second, hundredths of a second, and thousandths of a second.
>9	Hours, minutes, seconds, tenths of a second, hundredths of a second, thousandths of a second, and padded with blanks.

QHOUR: QHOUR is the system value for the hour of the day. Hours are based on a 24-hour clock. For example, 1:00 p.m. is 13. A change to this system value takes effect immediately and affects the QTIME system value.

Type	Length	Shipped Value
Character	2	No shipped value

QMINUTE: QMINUTE is the system value for the minute of the hour. Its value can range from '00' through '59'. A change to this system value takes effect immediately, and changes the system value QTIME.

Type	Length	Shipped Value
Character	2	No shipped value

QSECOND: QSECOND is the system value for the second of the minute. Its value can range from '00' through '59'. A change to this system value takes effect immediately, and changes the system value QTIME.

Type	Length	Shipped Value
Character	2	No shipped value

Editing System Values

The editing system values allow you to control the format used for displaying currency symbols, dates, and decimal values on your system.

All changes to the editing system values take effect immediately for new jobs, and for the system values QCURSYM and QDECFMT, the changes also affect active jobs.

QCURSYM: QCURSYM is the system value for the currency symbol. This system value is used to validate the currency symbols specified in the DDS keywords EDTWRD and EDTCDE. QCURSYM can be any character except blank, -, &, *, or 0.

A change to this system value takes effect immediately and affects active jobs.

Type	Length	Shipped Value
Character	1	May be different for different countries.

QDATFMT: QDATFMT is the system date format. This system value can be YMD, MDY, DMY, or JUL (Julian format). (Y = year; M = month; D = day.) This system value is used for the following:

- The default value for the DATFMT job attribute
- To determine the format in which a date can be specified on the IPL options prompt

A change to this system value takes effect for new jobs that enter the system after the change.

Type	Length	Shipped Value
Character	3	May be different for different countries.

QDATSEP: QDATSEP is the character separator for dates. This system value is used as the date separator for the default value of the DATSEP job attribute or in the date that you can specify on the IPL options prompt.

QDATSEP can be '/', '-', '.' (period), ',' (comma), or ' ' (blank). The Work with System Values (WRKSYSVAL) and Display System Values (DSPSYSVAL) commands use numbers to represent each of the values for QDATSEP:

1 = '/'
2 = '-'
3 = '.'
4 = ','
5 = blank

The Change System Value (CHGSYSVAL) and Retrieve System Value (RTVSYSVAL) commands use the actual values for QDATSEP.

A change to this system value takes effect for new jobs that enter the system after the change.

Type	Length	Shipped Value
Character	1	May be different for different countries.

QDECFMT: QDECFMT is the decimal format. This system value is used for the following:

- To determine the type of zero suppression and decimal point character used by DDS edit codes 1 through 4 and A through M
- To determine the decimal point character for decimal input fields on displays

QDECFMT must be one of the following characters:

- ' ' (blank): Use a period for a decimal point, use a comma for a three-digit grouping character, and zero-suppress to the left of the decimal point. For example,

  ```
  1,000.04      .04
  ```

- J: Use a comma for a decimal point, and use a period for a three-digit grouping character. The zero-suppression character is in the second position (rather than the first) to the left of the decimal notation. Balances with zero values to the left of the comma are written with one leading zero (0,04). The J entry also overrides any edit codes that might suppress the leading zero. For example,

  ```
  1.000,04      0,04
  ```

- I: Use a comma for a decimal point, use a period for a three-digit grouping character, and zero-suppress to the left of the decimal point. For example,

  ```
  1.000,04      ,04
  ```

The Work with System Value (WRKSYSVAL) and Display System Value (DSPSYSVAL) commands use numbers to represent each of the values for QDECFMT:

1 = blank
2 = J
3 = I

The Change System Value (CHGSYSVAL) and Retrieve System Value (RTVSYSVAL) commands

use the actual values for QDECFMT: blank, J, or I.

A change to this system value takes effect immediately and affects active jobs.

Type	Length	Shipped Value
Character	1	May be different for different countries.

QTIMSEP: QTIMSEP is the character separator for time. This system value is used as the time separator for the default value of the TIMSEP job attribute or in the time you can specify on the IPL options prompt.

QTIMSEP must be one of the following values: ':'(colon), '.' (period), ',' (comma), or ' ' (blank).

The Work with System Value (WRKSYSVAL) and Display System Value (DSPSYSVAL) commands use numbers to represent each of the values for QTIMSEP:

1 = ':'
2 = '.'
3 = ','
4 = blank

The Change System Value (CHGSYSVAL) and Retrieve System Value (RTVSYSVAL) commands use the actual values for QTIMSEP.

A change to this system value takes effect for new jobs that enter the system after the change.

Type	Length	Shipped Value
Character	1	May be different for different countries.

System Control System Values

The system control system values allow you to control or display information specific to your system.

QABNORMSW: QABNORMSW is the previous end of system indicator. This value indicates if the previous end of system was normal or abnormal. The previous end was normal if it is the result of a successful PWRDWNSYS command and the ENDJOBABN command was not used. If you end a job using the ENDJOBABN command, the next system end will be abnormal. You cannot change QABNORMSW; it is set by the system. QABNORMSW can be:

- '0': Previous end of system was normal.
- '1': Previous end of system was abnormal.

You can refer to this value in user-written recovery programs.

Type	Length	Shipped Value
Character	1	'0'

QASTLVL: QASTLVL is the assistance level system value. The value specifies the level of assistance available to users of the system.

This system value is used to tailor the level of displays available for users of the system. Displays intended for less experienced users provide a higher level of assistance than do displays intended for expert users.

- '*BASIC': Operational Assistant level of system displays is available.
- '*INTERMED': Intermediate level of system displays is available.
- '*ADVANCED': Advanced level of system displays is available.

A change to this system value takes effect the next time a user signs on.

Type	Length	Shipped Value
Character	10	'*BASIC'

QATNPGM: QATNPGM is the attention program system value. The value specifies whether the Attention key calls the Operational Assistant main menu.

- '*ASSIST': The Operational Assistant main menu appears when the Attention key is pressed.
- '*NONE': No Attention program is called when the Attention key is pressed.
- 'PGM LIB': The name and library of the program to be called when the Attention key is pressed.

A change to this system value takes effect the next time a user signs on.

Note: The value *ASSIST has the same effect as specifying 'QEZMAIN QSYS'. If the value is *ASSIST, the Retrieve System Value (RTVSYSVAL) command retrieves 'QEZMAIN QSYS', even though the value is displayed as *ASSIST through the Work with System Value (WRKSYSVAL) command and the Display System Value (DSPSYSVAL) command.

Type	Length	Shipped Value
Character	20	'*ASSIST'

QAUTOCFG: QAUTOCFG system value automatically configures locally attached devices. The value specifies whether devices that are added to the system are configured automatically.

The system value QDEVNAMING controls the names the system uses when configuring automatically.

- '0': Automatic configuration is off. You have to manually configure any new local controllers or devices that you add to your system.
- '1': Automatic configuration is on. The system automatically configures any new local controllers or devices that are added to your system. The operator receives a message indicating the changes to the system's configuration.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Character	1	'1'

For information on configuring devices for SNA Primary LU 2 support, see the *Remote Work Station Guide*.

QAUTOVRT: QAUTOVRT is the system value for automatic configuration of virtual devices. This is a decimal system value with which the user specifies a number in the range of 0 through 9999, which is the number of virtual devices that the user wants to have automatically configured. If the user does not want pass-through to automatically configure any devices, the value should be 0. The virtual devices are deleted only if they are damaged, or if the device needs to be created again to change its type. Devices are not automatically deleted to bring the total number down to the QAUTOVRT limit. Therefore, if the user changes from a higher value to a lower value, and if they expect that lower value to represent the real maximum number of virtual devices, the system does not delete virtual devices.

Note: Be aware of the following security considerations with this system value: The number of sign-on attempts allowed at remote devices increases when QAUTOVRT is greater than 0. The number of attempts allowed through pass-through is QAUTOVRT multiplied by the maximum number of sign-on attempts (the system value QMAXSIGN).

Type	Length	Shipped Value
Decimal	(5 0)	0

QCCSID: QCCSID is the system value for coded character set identifiers, which identify:

- A specific set of encoding scheme identifiers
- Character set identifiers
- Code page identifiers
- Additional coding-related information that uniquely identifies the coded graphic character representation to be used by the system

QCCSID should be set based on the language installed on the system. On a system capable of using double-byte character sets (DBCS), QCCSID must be set to a mixed CCSID (a CCSID that represents both single- and double-byte character set and code page). On a system that is not capable of using DBCS, QCCSID must be set to a single-byte character set (SBCS) CCSID. The QIGC system value tells if a system is capable of using DBCS.

A change to the QCCSID system value may automatically change the QCHRID system value to maintain compatibility between the two system values. The *National Language Support Planning Guide* contains more information on coded character set identifiers.

Type	Length	Shipped Value
Decimal	(10 0)	'65535'

QCHRID: QCHRID is the default character set and code page. This system value specifies the character set and code page to be used when CHRID(*SYSVAL) is specified for the CL commands that create, change, or override display files, display device descriptions, and printer files. The value you specify should be one of the values shown for the CHRID parameter on the CRTDSPF, CRTDEVD, or CRTPRTF command. Specify a value that reflects the language used at most of the devices attached to your system. The *Data Management Guide* contains more information in the sections on using alternative graphic character sets and code pages.

If the QCCSID system value is something other than 65535, a change to QCHRID must be compatible with the value of QCCSID. If the value specified for QCHRID is not compatible with QCCSID, the change is not allowed. See the *National Language Support Planning Guide* for more information.

QCHRID is a single system value with two parts:

- Character set identifier: This part identifies the character set to use. This identifier must be in the range of 1 through 32767.
- Code page identifier: This part identifies the code page to use. This identifier must be in the range of 1 through 32767.

The QCHRID system value is retrieved as a single character value; the first 10 characters contain the character set identifier right-adjusted. For example, the value 101 would be retrieved as 0000000101. The last 10 characters contain the code page identifier right-adjusted. For example, the value 37 would be retrieved as 0000000037. The QCHRID system value is displayed in the same format as other list type system values.

Changes take effect immediately for newly created display files, display device descriptions, and printer files.

Type	Length	Shipped Value
Character	20	May be different for different countries.

QCMNRCYLMT: QCMNRCYLMT is the system value for communications recovery limits. QCMNRCYLMT is a single system value with two parts:

- Count limit: The number of recovery attempts (0 through 99) to be made by the system before an inquiry message is sent to the system operator.
- Time interval: The time period (0 through 120 minutes) before the system will send an inquiry message to the system operator if the count limit is reached.

Count Limit	Time Interval	Action
0	0	No recovery
0	1 through 120	No recovery
1 through 99	0	Infinite recovery
1 through 99	1 through 120	Count and time recovery

If your AS/400 system is attached to a ROLM** CBX, the recovery attempts value should never be 0. Recovery attempts are necessary for the AS/400 system to establish a connection using the ROLM CBX's inbound modem pool.

A change to the QCMNRCYLMT system value does not affect a varied on device, but is in effect when a device is varied on. The *Communications Management Guide* contains more information on communications recovery.

Type	Length	Shipped Value
Character	20	'0 0' (0 count limit and 0 time interval, which means that no recovery attempts will be made).

The QCMNRCYLMT system value is retrieved as a 20-character value; the first 10 characters contain the count limit right adjusted. For example, the value 7 would be retrieved as '0000000007'. The last 10 characters contain the time interval right adjusted. For example, the value 117 would be retrieved as '0000000117'. Both values are integer values and must be specified in a quoted string. A blank must separate the count limit from the time interval. The QCMNRCYLMT is displayed in the same format as other list type system values.

QCNTRYID: QCNTRYID is the system value for the country identifier. This value specifies the country identifier to be used as the default on the system. For more information, see the *National Language Support Planning Guide*.

Type	Length	Shipped Value
Character	2	May be different for different countries

QCONSOLE: QCONSOLE is the console name. This value specifies the name of the display device that is the console. You cannot change this system value. The system changes this value when the console is varied on.

Type	Length	Shipped Value
Character	10	'QCONSOLE'

QCTLSBSD: QCTLSBSD is the controlling subsystem description. The controlling subsystem is the first subsystem to start after an initial program load (IPL). One subsystem must be active while the system is running. This is the controlling subsystem. Other subsystems can be started and stopped.

The subsystem you sign on to is displayed in the upper right corner of the sign-on prompt. During an IPL, if the QCTLSBSD system value is changed by selecting Define or Change System on the IPL options display, the user is signed on a different subsystem than the one that was displayed on the sign-on display. This is because the change takes effect immediately.

A change to this value takes effect at the next IPL unless the change is made on the IPL options display during an IPL. If that is the case, a change to this value takes effect immediately. If this subsystem description cannot be used (for example, it is damaged), the backup subsystem description QSYSSBSD in the library QSYS can be used. A subsystem description specified as the controlling subsystem cannot be deleted or renamed once the system is fully operational. IBM ships three controlling subsystem descriptions that can be used. See Appendix C, "Characteristics of the Shipped System" on page C-1 for more information about the shipped subsystem descriptions. See "Creating Another Controlling Subsystem" on page 3-21 or "Changing the Controlling Subsystem" on page 3-21 for other items to consider before changing the value of QCTLSBSD.

Type	Length	Shipped Value
Character	20	'QBASE QSYS'

The value of QCTLSBSD is a 20-character list of up to two 10-character values in which the first is the subsystem description name and the second is the library name. The list must be specified as a quoted string separating the subsystem description name and the library name by a blank if the library is specified. Apostrophes are necessary only when the library name or *LIBL is specified with the subsystem name. If the library name is not specified or *LIBL is specified for the library, the library list is used to locate the subsystem description, and the library where the subsystem description is found is stored in the system value. If *LIBL is specified for the library name for a change made from selecting Define or Change System at IPL Menu on the IPL options display, during an IPL, QSYS is the only library in the library list (*LIBL).

QDBRCVYWT: QDBRCVYWT is the database recovery wait indicator. This system value controls when the recovery of database files created with the RECOVER(*AFTIPL) option is performed during an unattended IPL. QDBRCVYWT can be:

- '0': Do not wait for database recovery to complete before completing the IPL. The database recovery after an abnormal end of system (QABNORMSW system value) can take awhile to complete. If you do not want to wait for it to complete before the system becomes available, choose this value.
- '1': Wait for database recovery to complete before completing the IPL. Files that were created with the RECOVER(*AFTIPL) option are treated as if they were created with the RECOVER(*IPL) option. This value does not affect recovery of files that were created with RECOVER(*IPL) or RECOVER(*NO).

A change to this value takes effect during the next IPL in unattended mode. QDBRCVYWT has no effect during an attended IPL. This value is related to the RECOVER parameter on the Create

Physical File (CRTPF) and Create Logical File (CRTLF) commands.

Type	Length	Shipped Value
Character	1	'0'

QDEVNAMING: QDEVNAMING is the naming convention for locally attached devices. This value specifies what naming convention is used when the system automatically creates device descriptions. You can change, display, and retrieve this value. QDEVNAMING must be one of the following values:

- '*NORMAL': Naming conventions should follow AS/400 standards.
- '*S36': Naming conventions should follow System/36 standards.
- '*DEVADR': Derive device name from the device address.

Note: You must have *ALLOBJ and *SECADM special authority to change the QDEVNAMING system value.

The following shows an example of the naming conventions:

Device	Normal	S36	DEVADR
Display Stations	DSP01, DSP02	W1, W2	DSP010403
Printer	PRT01, PRT02	P1, P2	PRT010302
Diskette Drive	DKT01	I1	DKT01
Tape Device	TAP01	T1	TAP01

Type	Length	Shipped Value
Character	10	'*NORMAL'

For more information on the device naming conventions, see the *Device Configuration Guide*. For information on device naming for SNA Primary LU2 support, see *Remote Work Station Guide*.

QDEVRCYACN: QDEVRCYACN specifies what action to take when an I/O error occurs for an interactive job's work station.

The values for QDEVRCYACN are:

- *MSG: Signals the I/O error message to the user's application program. The application program performs error recovery.
- *DSCENDRQS: Disconnects the job. When signing-on again, a cancel request function is performed to return control of the job back to the last request level.
- *DSCMSG: Disconnects the job. When signing-on again, an error message is sent to the user's application program.
- *ENDJOB: Ends the job. A job log is produced for the job. A message indicating that the job ended because of the device error is sent to the job log and the QHST log. To minimize the performance impact of the ending job, the job's priority is lowered by 10, the time slice is set to 100 milliseconds and the purge attribute is set to yes.
- *ENDJOBNOLIST: Ends the job. A job log is not produced for the job. A message is sent to the QHST log indicating that the job ended because of the device error.

Information on jobs ending at the same time can be found on page 5-9.

A change to this system value takes effect for new jobs started after the change. This system value should be reviewed and possibly changed from the shipped value of *MSG when AS/400 systems are operating in a large LAN/WAN environment.

Type	Length	Shipped Value
Character	20	'*MSG'

QDSCJOBITV: QDSCJOBITV indicates the length of time, in minutes, an interactive job can be disconnected before it is ended. A job can be disconnected using the DSCJOB command or when an I/O error occurs at the interactive job's work station (the system value QDEVRCYACN). If the time interval is exceeded, the disconnected job ends.

Disconnected jobs end abnormally at IPL.

The values for QDSCJOBITV are:

- '5' through '1440': The range of the disconnect interval.
- *NONE: There is no disconnect interval.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Character	10	'240'

QIGC: QIGC is the DBCS version indicator. This value specifies if the DBCS version of the system is installed. You cannot change QIGC; it is set by the system. You can refer to this system value in an application program. QIGC can be:

- '0': A DBCS version is not installed.
- '1': A DBCS version is installed.

Type	Length	Shipped Value
Character	1	No shipped value

QIGCCDEFNT: QIGCCDEFNT is the system value for the double-byte coded font name. It is used when transforming SNA character string (SCS) data into an advanced function printing data stream (AFPDS) spooled file with shift in/shift out (SI/SO) characters present in the data.

QIGCCDEFNT is a 20-character list of up to two values in which the first 10 characters contain the coded font name and the last 10 characters contain the library name. The possible values for the DBCS coded font name are:

- '*NONE': No coded font is identified to the system.
- '*coded font name*': The name of the DBCS coded font.

The possible values for the library are:

- '*LIBL': The library list is used to locate the coded font.
- '*CURLIB': The current library is used to locate the coded font. If no library is specified, library QGPL is used.
- '*library name*': The library containing the coded font.

Note: The coded font name can be only 8 characters.

Type	Length	Shipped Value
Character	20	May be different for different countries.

QIPLDATTIM: QIPLDATTIM is the system value for the date and time to automatically IPL the system. It specifies a date and time when an automatic IPL should occur. The default value *NONE indicates that no timed automatic IPL is desired. QIPLDATTIM is a single system value with two parts:

- Date: The date an IPL automatically occurs on the system. The date cannot be more than 11 months after the current date. The date is in QDATFMT format with no date separators.
- Time: The time an IPL automatically occurs on the system. The seconds portion of the time value must be specified, but it is ignored. The time is specified with no separators.

Note: If the date and time have already occurred when the system is powered down or the system is running when the date and time occur, no IPL is performed. After the date/time IPL occurs, the value of QIPLDATTIM is changed to *NONE. It is not changed to *NONE unless the IPL occurs.

The following example shows how to change the IPL date and time to September 10, 1990 (QDATFMT is MDY) at 9:00 a.m..

```
CHGSYSVAL SYSVAL(QIPLDATTIM) VALUE('091090 090000')
```

Note: You must have *ALLOBJ and *SECADM special authority to change the QIPLDATTIM system value.

Type	Length	Shipped Value
Character	20	'*NONE'

QIPLSTS: QIPLSTS is the IPL status indicator. This value indicates what form of IPL has occurred. Each form of IPL has an indicator value. You can refer to this value in your recovery programs, but you cannot change it.

- '0': Operator panel IPL. IPL occurred when requested from the operator panel.
- '1': Automatic IPL after power restored. IPL occurred automatically when power was restored after a power failure. This is enabled by the QPWRRSTIPL system value.

- '2': Restart IPL. IPL occurred when the Power Down System (PWRDWNSYS) command with RESTART(*YES) was issued.
- '3': Time-of-day IPL. IPL occurred automatically on the date and time set on the QIPLDATTIM system value.
- '4': Remote IPL. Remote IPL occurred. This is enabled by the QRMTIPL system value.

Type	Length	Shipped Value
Character	1	'0'

QIPLTYPE: QIPLTYPE indicates the type of IPL to perform. This value specifies the type of IPL performed when the system is powered on manually with the key in the normal position. This value is used only when the key is in normal position. QIPLTYPE can be:

- '0': Unattended. No displays are shown during an IPL. The normal sign-on display is shown when the IPL is complete.
- '1': Attended with dedicated service tools. All dedicated service tools functions are available along with the full set of IPL displays. If the system is in manual mode, '0' changes to '1'.
- '2': Attended with console in debug mode. This leaves the controller QCTL and device QCONSOLE varied on after an IPL. *You should only use this for problem analysis, as it prevents other devices on the work station controller from being used.*

Note: You must have *ALLOBJ and *SECADM special authority to change the QIPLTYPE system value.

Type	Length	Shipped Value
Character	1	'0'

QKBDBUF: QKBDBUF specifies whether the type-ahead feature and buffer Attention key option should be used.

- '*TYPEAHEAD': The type-ahead feature is turned on and the Attention key buffering option is turned off.
- '*NO': The type-ahead feature and the Attention key buffering option are turned off.
- '*YES': The type-ahead feature and the Attention key buffering option are turned on.

Note: You must have *ALLOBJ and *SECADM special authority to change the QKBDBUF system value.

A change to this system value takes effect the next time the user signs on.

Type	Length	Shipped Value
Character	10	'*TYPEAHEAD'

QKBDTYPE: QKBDTYPE specifies the language character set for the keyboard. Based on the language value specified at install time, the OS/400 licensed program will put the appropriate keyboard value into the system value. QKBDTYPE can be:

QKBDTYPE

Value	Language or Country
AGB	Austria/Germany
AGI	Austria/Germany Multinational
BLB	Belgium
BLI	Belgium Multinational
BRB	Brazilian Portuguese
CAB	Canadian French
CAI	Canadian French Multinational
CLB	Arabic X
CYB	Cyrillic
DMB	Denmark
DMI	Denmark Multinational
FNB	Finland/Sweden
FNI	Finland/Sweden Multinational
FAB	France (Azerty)
FAI	France (Azerty) Multinational
FQB	France (Qwerty)
FQI	France (Qwerty) Multinational
GKB	Greek
GNB	Greek
ICB	Iceland
ICI	Iceland Multinational
INB	International
INI	International Multinational
ITB	Italy
ITI	Italy Multinational
JEB	Japan (English)
JKB	Japan (Kanji) and Katakana
JUB	Japanese (Kanji) and US English
KAB	Japan (Katakana)
KOB	Korea
NCB	Hebrew
NEB	Netherlands
NEI	Netherlands Multinational
NWB	Norway
NWI	Norway Multinational

QKBDTYPE

Value	Language or Country
PRB	Portugal
PRI	Portugal Multinational
RCB	Simplified Chinese
ROB	Latin 2
SPB	Spain
SPI	Spain Multinational
SSB	Spanish Speaking
SSI	Spanish Speaking Multinational
SWB	Sweden
SWI	Sweden Multinational
SFI	Switzerland/French Multinational
SGI	Switzerland/German Multinational
TAB	Traditional Chinese
THB	Thailand
TKB	Turkey
UKB	United Kingdom
UKI	United Kingdom Multinational
USB	United States/Canada
USI	United States/Canada Multinational
YGI	Languages of the Former Yugoslavia

Type	Length	Shipped Value
Character	3	'USB' (for United States and Canada)

QLANGID: QLANGID is the system value for the language identifier. This system value specifies the language identifier to be used as the default for the system. If you specify *LANGIDSHR or *LANGIDUNQ for the SRTSEQ parameter, the sort sequence table used is the unique weight sort table or shared weight sort table associated with QLANGID system value.

For more information, see the *National Language Support Planning Guide*.

Type	Length	Shipped Value
Character	3	May be different for different countries

QMODEL: QMODEL is the system model number. You cannot change QMODEL, but the four-character value can be displayed or retrieved in user written programs. Examples of values you may see are: B10, B20, B35, or B70.

Type	Length	Shipped Value
Character	4	No shipped value

QPFRADJ: QPFRADJ is the performance adjustment. This value indicates whether the system should adjust values during IPL and dynamically for system pool sizes and activity levels.

- '0': No performance adjustment. The existing values for system pool sizes and activity levels are used.
- '1': Performance adjustment at IPL. The values for system pool sizes and activity levels are calculated and changed during IPL. The following values are changed:
  - QMCHPOOL system value
  - QBASACTLVL system value (if QSYS/QBASE, QSYS/QCTL, QGPL/QBASE, or QGPL/QCTL is the controlling subsystem)
  - Pool 2 of QGPL/QSPL subsystem description
  - Pool 2 of QSYS/QBASE subsystem description (if QSYS/QBASE is controlling subsystem)
  - Pool 2 of QGPL/QBASE subsystem description (if QGPL/QBASE is controlling subsystem)
  - Pool 2 of QGPL/QINTER subsystem description (if QSYS/QCTL or QGPL/QCTL is controlling subsystem)
- '2': Performance adjustment at IPL and dynamically. The values for system pool sizes and activity levels are calculated and changed during IPL and at regular intervals thereafter. The following values are changed:
  - QMCHPOOL system value
  - QBASACTLVL system value
  - Pool 2 of QGPL/QSPL subsystem description
  - Pool 2 of QSYS/QBASE subsystem description (if QSYS/QBASE is controlling subsystem)
  - Pool 2 of QGPL/QBASE subsystem description (if QGPL/QBASE is controlling subsystem)
  - Pool 2 of QGPL/QINTER subsystem description (if QSYS/QCTL or QGPL/QCTL is controlling subsystem)

  - *INTERACT, *BASE, *SPOOL, *SHRPOOL1–*SHRPOOL10 pool sizes and activity levels
- '3': Dynamic performance adjustment. The values for system pool sizes and activity levels are calculated and changed at regular intervals. The following values are changed:
  - QMCHPOOL system value
  - QBASACTLVL system value
  - *INTERACT, *BASE, *SPOOL, *SHRPOOL1–*SHRPOOL10 pool sizes and activity levels

The value of QPFRADJ should be set to '0' if you have set any of these values.

Note: If you choose to manually tune your system, do not change '0' to '1,' '2,' or '3.'

| Changes to this system value take effect immediately.

Type	Length	Shipped Value
Character	1	'2'

QPRTDEV: QPRTDEV is the default printer device description. This value specifies the default printer for the system, and takes effect when a spooled file is opened. If the specified device description exists, it must be a printer device description.

A change to this system value takes effect for new jobs started in the system.

Note: You must have *ALLOBJ and *SECADM special authority to change the QPRTDEV system value.

Type	Length	Shipped Value
Character	10	'PRT01'

QPRTKEYFMT: QPRTKEYFMT is the print key format. This value specifies whether border and header information is provided when the Print key is pressed.

A change to this system value takes effect for new jobs started on the system.

QPRTKEYFMT can be:

- *NONE: The border and header information is not included with output from the Print key.
- *PRTBDR: The border information is included with output from the Print key.
- *PRTHDR: The header information is included with output from the Print key.
- *PRTALL: The border and header information is included with output from the Print key.

Type	Length	Shipped Value
Character	10	'*PRTHDR'

QPWRDWNLMT: QPWRDWNLMT is the maximum amount of time an immediate power down can take before processing is ended (abnormal end). This is the time used to wait for power down to complete normally after either of the following happens:

- A Power Down System (PWRDWNSYS) command with the OPTION(*IMMED) parameter is entered.
- A PWRDWNSYS command with the OPTION(*CNTRLD) parameter entered and the time specified on the DELAY parameter has ended.

The QPWRDWNLMT value is ignored when either of these commands is entered after a power failure has occurred on a system with the power warning feature installed. (See the *Advanced Backup and Recovery Guide* for more details.)

A change to this value takes effect when a PWRDWNSYS command is entered. QPWRDWNLMT is numeric. The lower limit for QPWRDWNLMT is 0. If the value is set to 0 and a PWRDWNSYS command with OPTION(*IMMED) is entered, processing is ended immediately (abnormal end).

Note: If the value is set to 0 (or a very small value), a power-down time-out condition occurs, and the system does not finish the power-down operation even though the system processing has ended.

Type	Length	Shipped Value
Decimal	(5 0)	600 seconds

QPWRRSTIPL: QPWRRSTIPL is the system value for an automatic IPL after power is restored. The value specifies if the system should automatically IPL when utility power is restored after a power failure. A change to this system value takes effect after the next power failure. QPWRRSTIPL can be:

- '0': Automatic IPL is not allowed.
- '1': Automatic IPL is allowed.

Note: You must have *ALLOBJ and *SECADM special authority to change the QPWRRSTIPL system value.

Type	Length	Shipped Value
Character	1	'0'

QRMTIPL: QRMTIPL is the remote power on and IPL indicator. This value specifies if remote power on and IPL can be started over a telephone line. QRMTIPL can be:

- '0': Remote power on and IPL are not allowed.
- '1': Remote power on and IPL are allowed.

Notes:

1. Any telephone call causes the system to IPL.
2. You must have *ALLOBJ and *SECADM special authority to change the QRMTIPL system value.

Type	Length	Shipped Value
Character	1	'0'

QSCPFCONS: QSCPFCONS is the IPL action with console problem indicator. This value specifies whether the IPL is to continue unattended or ends when the console is not operational when performing an attended IPL. QSCPFCONS must have been set before the current IPL. QSCPFCONS can be:

- '0': End system. QSCPFCONS should be 0 if work stations other than the console are not on the system or if the controlling subsystem supports only the console and does not start other subsystems that support other work stations.
- '1': Continue the IPL unattended.

Type	Length	Shipped Value
Character	1	'1'

QSPCENV: QSPCENV is the special environment indicator. This value specifies the system environment used as the default for all users. QSPCENV can be:

- '*NONE': You enter the AS/400 system environment when you sign on.
- '*S36': You enter the System/36 environment when you sign on.

A change to this system value takes effect the next time a user signs on.

Note: You must have *ALLOBJ and *SECADM special authority to change the QSPCENV system value.

Type	Length	Shipped Value
Character	10	'*NONE'

QSRLNBR: QSRLNBR is the system serial number. You cannot change QSRLNBR; it is retrieved from the data fields by the system when installing the OS/400 licensed program. You can display QSRLNBR, or you can retrieve this value in user-written programs. An example of a serial number is 1001003.

Type	Length	Shipped Value
Character	8	No shipped value

QSRTSEQ: QSRTSEQ specifies the default sort sequence to be used by the system. The following are possible values:

- *HEX: No sort sequence table is used. The hexadecimal values of the characters are used to determine the sort sequence.
- *LANGIDSHR: The sort sequence table used can contain the same weight for multiple characters. It is the shared weight sort table associated with the language specified in the LANGID parameter.
- *LANGIDUNQ: The sort sequence table used must contain a unique weight for each character in the code page. It is the unique weight

sort table associated with the language specified in the LANGID parameter.

- *Qualified sort sequence table name*: The name and library of the sort sequence table to be used.

Type	Length	Shipped Value
Character	20	'*HEX'

QSTRUPPGM: QSTRUPPGM is the start-up program. This value specifies the name of the program called from an autostart job when the controlling subsystem is started. This program performs setup functions, such as starting subsystems and printers. This system value can only be changed by the security officer or by someone with security officer authority. A change to this system value takes effect the next time an IPL is performed. QSTRUPPGM can have the values:

- 'QSTRUP QSYS': The program specified is run as a result of a transfer of control to it from the autostart job in the controlling subsystem.
- '*NONE': The autostart job ends normally without calling a program.

The default startup program QSYS QSTRUP does the following:

- Starts the QSPL subsystem for spooled work.
- Releases the QS36MRT and QS36EVOKE job queues if they were held (these are used by the System/36 environment).
- Starts Operational Assistant cleanup, if allowed.
- Starts all print writers unless user specified not to on the IPL Options display.
- If the controlling subsystem is QCTL, it starts the QINTER, QBATCH, and QCMN subsystems.

Type	Length	Shipped Value
Character	20	'QSTRUP QSYS'

QSTRPRTWTR: QSTRPRTWTR specifies whether printer writers are started at IPL. This system value is either set by the system at IPL time or by the user on the IPL Options display. The IBM-supplied startup program QSTRUP uses this system value to determine whether to start printer writers. QSTRPRTWTR can be:

- '0': Do not start printer writers.
- '1': Start printer writers.

This value can only be displayed or retrieved.

Type	Length	Shipped Value
Character	1	'1'

QUPSDLYTIM: QUPSDLYTIM is the uninterruptible power supply delay time. The QUPSDLYTIM system value indicates the amount of time that should elapse before the system automatically powers down following a power failure. When a change in power activates the uninterruptible power supply, messages are sent to the uninterruptible power supply message queue (the system value QUPSMSGQ). This system value is only meaningful if your system has the battery power unit or has an uninterruptible power supply attached. The allowed values are:

Value	Description
'*BASIC'	Powers only the PRC, IOP cards, and load source storage. The appropriate wait time, in seconds, is calculated. (This should only be used if you have the battery power unit or an uninterruptible power supply without every rack being connected.) **Notes:** 1. All other values indicate an uninterruptible power supply on all racks. 2. This value should not be used for a 9402 or 9404 system. 3. The calculated value may have to be adjusted by the user if there has been a series of power outages, or if the batteries are not fully charged.
'*CALC'	The appropriate wait time (in seconds) is calculated. This value should only be used if you have a 9402 or 9404 system with a battery power unit.

Value	Description
	Notes: 1. The calculated value is appropriate for the 9402 and 9404 user (although there is no restriction on its use by a 9406 user). 2. The calculated value is based on the amount of devices and the number of changed pages in main storage. 3. The calculated value may have to be adjusted by the user if there has been a series of power outages, or if the batteries are not fully charged.
'*NOMAX'	The system does not start any action on its own. **Note:** If you use '*NOMAX' you must also meet the requirements of QUPSMSGQ (see page 2-26 for these requirements).
'0'	Automatic system power down when system utility power fails.
'1'-'99999'	Delay time specified in seconds before the system powers down.

A change to this system value takes effect the next time there is a power failure. For more information on the uninterruptible power supply feature, see the *Advanced Backup and Recovery Guide*.

Type	Length	Shipped Value
Character	20	'*CALC'

The QUPSDLYTIM system value is in the form of a two-item list. The first item is the value the user specified on the CHGSYSVAL command. The second item is the delay time, which is either what the user specified or, if *CALC or *BASIC is specified, is the calculated delay time.

You specify only the first item in the list. The second item, if specified, will be ignored. If you want to retrieve the QUPSDLYTIM system value, you would retrieve it into a CHAR 20 variable. The first 10 characters would be what you specified. The second 10 characters would be the delay time value (calculated by the machine or specified by you) you would use in your program.

10 Characters	10 Characters
0000000340	0000000340
*NOMAX	*NOMAX
*CALC	0000000120
*BASIC	0000000150

The following examples show what the user could enter and how the value would be displayed:

Example 1 '*CALC'

User types:

```
CHGSYSVAL QUPSDLYTIM *CALC
```

Using DSPSYSVAL, display shows:

```
Delay Time . . . . . :  *CALC
Delay value in seconds:  60
```

Example 2 '*NOMAX'

User types:

```
CHGSYSVAL QUPSDLYTIM *NOMAX
```

Display shows:

```
Delay Time . . . . . :  *NOMAX
Delay value in seconds:  *NOMAX
```

Example 3 '*BASIC'

User enters:

```
CHGSYSVAL QUPSDLYTIM *BASIC
```

```
Display shows:
Delay Time . . . . . :  *BASIC
Delay value in seconds:  150
```

QUPSMSGQ: Name of a message queue that is to receive uninterruptible power supply messages. If the message queue is not the system operator message queue (QSYSOPR QSYS), then all uninterruptible power supply messages are also sent to the QSYSOPR message queue. This system value is meaningful only if your system has the battery power unit feature and has an uninterruptible power supply attached.

When a change in power activates the uninterruptible power supply, this message queue receives the uninterruptible power supply activated message (CPF1816). If the QUPSDLYTIM is '*NOMAX', the following conditions must be met or the system begins an immediate power down.

- The message queue specified in the QUPSMSGQ system value must exist.
- If the message queue is a work station message queue (or QSYSOPR), it must be in break or notify mode.
- If the message queue is not a work station message queue, it must be allocated by a job.

For all other uninterruptible power supply messages, the message queue does not have to be allocated, or in break or notify mode. If the system value QUPSMSGQ does not contain the name of a valid message queue, a message is be sent to QSYSOPR indicating the notification failure, and the system continues processing.

For more information on the uninterruptible power supply feature, see the *Advanced Backup and Recovery Guide*.

A change to this system value takes effect the next time there is a power failure.

Type	Length	Shipped Value
Character	20	'QSYSOPR QSYS'

Library List System Values

The library list system values allow you to control or display the system and user parts of the library list.

QSYSLIBL: QSYSLIBL is the system part of the library list. The list can contain as many as 15 names. When searching for an object in the library list, the libraries in the system part are searched before any libraries in the user part are searched. A library specified as part of the library list cannot be deleted or renamed once the system is fully operational.

Note: To change the QSYSLIBL system value, you must have *ALLOBJ and *SECADM special authority, which can be from a user profile, a group profile, or it can be adopted by a program.

To change the QSYSLIBL system value, QSYS must be one of the libraries specified.

A change to this value takes effect when the next job starts.

Type	Length	Shipped Value
Character	150	'QSYS QSYS2 QHLPSYS QUSRSYS'

Its value is a 150-character list, enclosed in apostrophes, of library names with each name up to 10 characters in length. When changing this system value using the CHGSYSVAL command, each name must be separated by a blank. Apostrophes are necessary only when more than one library name is specified (for example, 'SYSLIB1 SYSLIB2').

QUSRLIBL: QUSRLIBL is the default for the user part of the library list. The list can contain as many as 25 names. The libraries in this part are searched for an object after the libraries in the system part and also after the product library and current library entries. A library specified as part of the library list cannot be deleted or renamed once the system is fully operational.

A change to this value takes effect when the next job starts.

Type	Length	Shipped Value
Character	250	'QGPL QTEMP'

Its value is a 250-character list, enclosed within apostrophes, of library names with each name up to 10 characters in length. When changing this system value using the CHGSYSVAL command, each name must be separated by a blank. Apostrophes are necessary only when more than one library name is specified. For example, 'USERLIB1 USERLIB2'.

Allocation System Values

The allocation system values allow you to control the number of jobs and storage sizes on your system.

QTOTJOB: QTOTJOB specifies the initial number of jobs for which auxiliary storage is allocated during IPL. The number of jobs is the number supported by the system at any one time, which includes the jobs on job queues, active jobs (including system jobs), and jobs having output on output queues.

Notes:

1. The system value QJOBSPLA is measured in bytes.
2. Additional space is not allocated for spool jobs that are on the system during an IPL.

To find the number of total jobs in the system, view the jobs in the system field of the system status display (using the WRKSYSSTS command). This number should usually be kept within reason as it is a factor in the time to perform IPL and some internal searches. This may require periodic removal of jobs that have only job logs. The *CL Programmer's Guide* has a discussion of job logs and how to remove them for jobs that complete normally. As long as a job has one or more spooled output files known to the system, knowledge of the job remains in the system and counts toward the display system status value. A reasonable value to assign to QTOTJOB is two to seven times the value used for QACTJOB, depending on how often you clear your output queues.

Note: You should set this value high enough so it will not normally be exceeded by the total number of jobs.

A change to this value takes effect at the next IPL unless the change is made on the IBM-supplied configuration menu, in which case, it is used for the current IPL. QTOTJOB may be changed to a smaller value during IPL if the combination of allocation system values requires more auxiliary storage than is available to start the system. If this happens, you are notified.

Type	Length	Shipped Value
Decimal	(5 0)	30

QACTJOB: QACTJOB is the initial number of active jobs for which auxiliary storage is to be allocated during IPL. An active job is a job that has started running but not ended. The amount of auxiliary storage allocated for each active job is approximately 110KB. This storage is in addition to the storage allocated using the system value QTOTJOB.

A reasonable value to assign to QACTJOB is your estimate of the number of active jobs on a typically heavy-use day. This can be done by viewing the active jobs field on the active jobs display (WRKACTJOB command). Both user and system jobs are included in the count on this display, but only user jobs need to be considered when assigning a value to QACTJOB.

Note: You should set this value high enough so that storage does not need to be allocated for additional active jobs using the system value QADLACTJ.

A change to this value takes effect at the next IPL unless the change is made on the IBM-supplied configuration menu, in which case it is used for the current IPL. QACTJOB may be changed to a smaller value during IPL if the combination of allocation system values requires more auxiliary storage than is available to start the system. If this happens, you are notified.

Type	Length	Shipped Value
Decimal	(5 0)	20

QJOBSPLA: QJOBSPLA specifies the initial size of the spooling control block for a job. (There is one spooling control block for each job in the system.) The spooling control block records information about inline spooled files and output spooled files. This value primarily affects auxiliary storage requirements and has little effect on processing performance. The auxiliary storage is retained for every job known to the system.

The allocated area is made up of standard control information plus a separate set of control information for each spooled file. The default is 1536 bytes, which allows for about 3 output spooled files per job.

If your typical job uses more than the three output files and you are not concerned with an additional 2KB allocation per job, a good choice would be 3600. This allows for approximately seven or eight spooled output files per job. The next largest logical choice would be 7600, which would round up to 8KB. Set the QJOBSPLA value so that it rounds up to a 2KB, 4KB, or 8KB value. QJOBSPLA lower limit is 1024 bytes.

If the number of spooled files created by a job exceeds the initial allocation, the value assigned to the QADLSPLA system value is used to extend the control block.

A change to this value takes effect when a cold start is requested during the installing of the OS/400 licensed program. A cold start clears the job and output queues. If the system requires new job structures, the value is also used for the new job structures (the system value QADLTOTJ).

Type	Length	Shipped Value
Decimal	(5 0)	1536

QADLTOTJ: QADLTOTJ specifies the additional number of jobs for which auxiliary storage is to be allocated when the initial number of jobs (the system value QTOTJOB) is reached. This auxiliary storage is allocated whenever the number of jobs exceeds that for which storage has already been allocated. *The amount of storage allocated for

The IBM-supplied value of 10 is recommended for QADLTOTJ. Setting the number close to 1 will cause excessive interruption when many additional jobs are needed. The number should not usually be set too high because the time required to add additional storage should be minimized.

A change made to this value takes effect immediately. QADLTOTJ may be changed to a smaller value during IPL if the combination of allocation system values requires more auxiliary storage than is available to start the system. If this happens, you are notified.

Type	Length	Shipped Value
Decimal	(5 0)	10

QADLACTJ: QADLACTJ specifies the additional number of active jobs for which auxiliary storage is to be allocated when the initial number of active jobs (the system value QACTJOB) is reached. An active job is a job that has started running but has not ended. This auxiliary storage is allocated whenever the number of active jobs exceeds the storage which has already been allocated. The amount of storage allocated for each job is approximately 110KB.

The IBM-supplied value of 10 is recommended for QADLACTJ. Setting the number close to 1 can cause frequent interruptions when many additional jobs are needed. The number should not be set too high because the time required to add additional storage should be minimized.

A change to this value takes effect immediately. QADLACTJ may be changed to a smaller value during IPL if the combination of allocation system values requires more auxiliary storage than is available to start the system. If this happens, you are notified.

Type	Length	Shipped Value
Decimal	(5 0)	10

QADLSPLA: QADLSPLA specifies the additional storage to add to the spooling control block. This storage is allocated when the number of spooled files exceeds the initial storage assigned by the QJOBSPLA system value or the last extension from QADLSPLA. This value affects auxiliary storage, but also affects processing performance each time an extension is needed. For this reason, a relatively high default of 2048 bytes is used to allow an extension for approximately four or five spooled files. No additional bytes are added to the value you supply, but the value is rounded up (if needed) to the next 512-byte page boundary.

If you have sufficient auxiliary storage, a value of 4096 is a good choice to allow for approximately nine or ten spooled files per extension. The next logical choice would be 8192. Set the QADLSPLA value so that it is a multiple of 2048. QADLSPLA lower limit is 1024 bytes.

A change to this system value is used the next time an extension to the spooling control block is needed.

Type	Length	Shipped Value
Decimal	(5 0)	2048

QJOBMSGQFL: QJOBMSGQFL specifies how the system should handle the job message queue when it is considered full. The system value QJOBMSGQMX indicates when a job message queue is considered full.

A change to the QJOBMSGQFL value takes place immediately.

The allowed values are:

- *NOWRAP: Do not wrap the job message queue. If you specify *NOWRAP and the value specified in QJOBMSGQMX is reached, the job ends.
- *WRAP: Wrap the job message queue.
- *PRTWRAP: Wrap the job message queue and print the messages that are being overlaid because of wrapping.

Type	Length	Shipped Value
Character	10	*NOWRAP

QJOBMSGQMX: QJOBMSGQMX specifies the maximum size of a message queue in megabytes. When this maximum size is reached for any job message queue, that job message queue is considered full and the action specified by QJOBMSGQFL is taken to control the full message job queue.

This system value can have a value between 8 and 64 megabytes.

A change to this system value affects any jobs started after the change is made.

Type	Length	Shipped Value
Decimal	(5 0)	16MB

QJOBMSGQSZ: This system value no longer affects the operating system.

QJOBMSGQTL: This system value no longer affects the operating system.

QRCLSPLSTG: QRCLSPLSTG is reclaim spool storage system value. It allows for the automatic removal of empty spool database members. The values allowed are:

- *NOMAX: The maximum retention interval. All empty members are kept. You must use the Reclaim Spooled Storage (RCLSPLSTG) command to delete empty spooled members from the system.
- *NONE: No retention interval. All empty members are deleted.
- '1' through '366': Number of days empty spool database members are kept for new spooled file use. If the members are still empty after the specified number of days, they are deleted by the system.

Type	Length	Shipped Value
Character	10	'8'

Message-and-Logging System Values

The message-and-logging system values allow you to control the messages and how they are logged on your system.

QACGLVL: QACGLVL is the accounting level. The possible accounting level options are:

Accounting Option	Accounting Information
*NONE	No accounting information is written to a journal.
*JOB	Job resource use is written to a journal.
*PRINT	Spooled and nonspooled print file resource use is written to a journal.

Note: Both *JOB and *PRINT can be used at the same time. However, *NONE cannot be used with the other options.

When the QACGLVL system value is changed to options other than *NONE, the system accounting journal QSYS/QACGJRN must exist; if not, the change is rejected. The QACGLVL system value cannot be changed on the IBM-supplied configuration menu during IPL. A change to the QACGLVL system value does not affect jobs already on the system, but does affect new jobs that enter the system after the change. For more information on job accounting, see "Setting Up Job Accounting" on page 15-10.

Note: You must have *ALLOBJ and *SECADM special authority to change the QACGLVL system value.

Type	Length	Shipped Value
Character	80	'*NONE'

QACGLVL's value is an 80-character list of accounting options. Apostrophes are necessary only when more than one option is specified and each option must be separated by blanks.

Note: If you specify Yes to save job accounting information about completed printer output using the Operational Assistant user interface, the QACGLVL system value is changed to *PRINT or *JOB *PRINT if job accounting is already on. If you change the system value later to turn job accounting off, the completed printer output option on the Operational Assistant menu will not be available.

QHSTLOGSIZ: QHSTLOGSIZ is the maximum number of records for each version of the history log. When a version is full (the maximum has been reached), a new version is created. You can save the full (old) version and then delete it. QHSTLOGSIZ is numeric. The lower limit for QHSTLOGSIZ is 1. (See the *CL Programmer's Guide* for more information about QHST.)

A change to this system value takes effect when a new history log version is created. Although 1 is the lower limit for QHSTLOGSIZ, using a small number affects system performance because it causes many history versions to be created.

Type	Length	Shipped Value
Decimal	(5 0)	5000

QPRBFTR: QPRBFTR is the system value for the problem filter name. This system value specifies the name of the filter object used by the service activity manager when processing problems. QPRBFTR is a 20-character list of up to two 10-character values in which the first value is the problem filter name and the second is the library name. The possible values for the problem filter name are:

- '*NONE': No problem filter is in use.
- '*problem filter name*': The name of the problem filter to be used.

The possible values for the library name are:

- '*LIBL': Use the library list when locating the filter object.
- '*CURLIB': Use the current library when locating the filter object.
- '*library name*': The name of the library where the problem filter is located.

Type	Length	Shipped Value
Character	20	'*NONE'

QPRBHLDITV: QPRBHLDITV is the problem log entry hold interval. This system value allows you to specify the minimum number of days a problem is kept in the problem log. After this time interval, the problem can be deleted using the Delete Problem (DLTPRB) command. The time interval starts as soon as it is put into the log. The range for this system value is 0 through 999 days.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Decimal	(5 0)	30

QPRTTXT: QPRTTXT is the print text. This system value is used to print up to 30 characters of text on the bottom of listings and separator pages. For the text to be centered at the bottom of the list, it must be centered in the QPRTTXT value field. The QPRTTXT system value is used for system and subsystem monitor jobs and is the system-wide default print text for all other jobs. If *BLANK is entered, QPRTTXT is set to all blanks.

A change to this system value takes effect for jobs that start after it is changed.

Type	Length	Shipped Value
Character	30	'*BLANK'

QSFWERRLOG: QSFWERRLOG is the system value for the software error log. This system value specifies whether software errors should be logged by the system. A change to this system value takes effect immediately. The allowed values are:

- *LOG: Software errors are logged in the error log, a problem alert record (PAR) message is sent to the QSYSOPR message queue, and an entry is created in the Problem Log in READY status.
- *NOLOG: No logging occurs.

Type	Length	Shipped Value
Character	10	'*LOG'

QSRVDMP: QSRVDMP specifies whether service dumps for unmonitored escape messages are created. The values that are allowed are:

- *DMPUSRJOB: Service dumps are only created for user jobs, not system jobs. System jobs include the system arbiter, subsystem monitors, LU services process, spool readers and writers, and the SCPF job.
- *DMPSYSJOB: Service dumps will only be created for system jobs, not user jobs.
- *DMPALLJOB: Service dumps will be created for all jobs.
- *NONE: Do not request dumps in any jobs.

Note: The ability to control the creation of service dumps with this system value may have an effect on IBM's ability to respond to some system failures. The value *DMPALLJOB can be used to ensure that a service dump is created whenever a failing function requests one. The shipped value (*DMPUSRJOB) provides that service dumps are created only for user jobs.

Type	Length	Shipped Value
Character	10	'*DMPUSRJOB'

QSTSMSG: QSTSMSG is the system value for status messages. This system value specifies whether or not the status message is displayed. A change to this system value takes effect the next time someone signs-on the system. Active interactive jobs are not changed. The values allowed are:

- *NORMAL: Status messages are displayed.
- *NONE: Suppresses status messages from being displayed.

Type	Length	Shipped Value
Character	10	'*NORMAL'

Storage System Values

The storage system values allow you to control storage size and activity levels on your system.

QMCHPOOL: QMCHPOOL is the size of the machine storage pool. The machine storage pool contains highly shared machine and OS/400 licensed programs. You must be careful when changing the size for this storage pool because system performance may be impaired if the storage pool is too small. QMCHPOOL is specified in KB, and cannot be set to less than 256KB.

This minimum value of 256KB, which is enforced by the Change System Value (CHGSYSVAL) command, does not reflect the machine-enforced minimum value. The machine-enforced minimum value varies depending on the main storage size of the machine. The system automatically increases the actual size of the machine storage pool to the machine-enforced minimum value if the QMCHPOOL system value contains a smaller value.

If the system has increased the actual size of the machine storage pool, you can use the Work with System Status (WRKSYSSTS) command to determine the actual machine-enforced minimum value for the machine storage pool (pool 1).

Chapter 13, "Performance Tuning" on page 13-1 contains a formula for determining the size of the machine storage pool for your system. You should use the result of this calculation, rather than the machine-enforced minimum value, to set the QMCHPOOL system value when you tune the performance of your system.

This value may be changed by the IPL performance adjust support or the dynamic performance adjust support when the system value QPFRADJ is set to 1, 2, or 3. Refer to "QPFRADJ" on page 2-22 for more information.

A change to this value takes effect immediately.

Type	Length	Shipped Value
Decimal	(10 0)	Different for each machine

QBASPOOL: QBASPOOL is the minimum size of the base storage pool. The base pool contains all main storage not allocated by other pools. This pool is specified in the subsystem description as *BASE. On the Work with System Status (WRKSYSSTS) command, this is system pool 2.

A change to this value takes effect immediately. In some circumstances, a machine function may be using storage allocated to the base pool. If this is so, and if the change to this system value would reduce the allocation to less than 32KB plus the amount needed by the machine, the system value is changed immediately (but the actual base pool change is done by the OS/400 licensed program only after the storage in use is released by the machine). Use the Work with System Status (WRKSYSSTS) command to determine the amount of storage reserved for machine functions. QBASPOOL is numeric, must be specified in KB, and cannot be set to less than 32KB.

Type	Length	Shipped Value
Decimal	(10 0)	500KB

QBASACTLVL: QBASACTLVL is the base storage pool activity level. This value indicates how many system and user jobs can compete at the same time for storage in the base storage pool. This pool is specified in the subsystem descriptions as *BASE. On the Work with System Status (WRKSYSSTS) command, this is system pool 2. QBASACTLVL depends on the types of jobs being run in this storage pool. The lower limit for QBASACTLVL is 1.

This value may be changed by the IPL performance adjust support or the dynamic tuning support when the system value QPFRADJ is set to 1, 2, or 3. Refer to "QPFRADJ" on page 2-22 for more information.

A change to this value takes effect immediately.

Type	Length	Shipped Value
Decimal	(5 0)	Different for each machine

QMAXACTLVL: QMAXACTLVL is the maximum activity level of the system. This is the number of jobs that can compete at the same time for main storage and processor resources. For all active subsystems, the sum of all jobs running in all storage pools cannot exceed QMAXACTLVL. If a job cannot be processed because the activity level has been reached, the job is held until another job reaches a time slice or a long wait. QMAXACTLVL is numeric. The lower limit for QMAXACTLVL is 2. A change to this value takes effect immediately. This value should be set to *NOMAX and the activity level should be controlled on a pool basis.

Note: The value *NOMAX has the same effect as specifying 32767. If the value is *NOMAX, the Retrieve System Value (RTVSYSVAL) command retrieves 32767 even though the value is displayed as *NOMAX through the Work with System Value (WRKSYSVAL) and the Display System Value (DSPSYSVAL) commands.

Type	Length	Shipped Value
Decimal	(5 0)	'*NOMAX'

QTSEPOOL: QTSEPOOL is the time slice end pool. This value specifies whether interactive jobs should be moved to another main storage pool when they reach time slice end. The job is moved back to the pool it was originally running in when a long wait occurs. This may help minimize the effect on interactive response time of other interactive jobs when one interactive job is performing a long running function. The values allowed are:

- *NONE: Jobs are not moved to the base storage pool when time slice end is reached.
- *BASE: Jobs are moved to the base pool when time slice end is reached.

If you allowed the IPL performance adjustment to tune the system, then you should set this value to *BASE.

A change to this value takes effect when a job is started. Active jobs are not changed.

Type	Length	Shipped Value
Character	10	'*NONE'

Security System Values

The security system values allow you to control security measures on your system.

To change these system values the user must have *ALLOBJ and *SECADM special authority, which can be from a user profile, group profile, or it can be adopted by a program. The auditing system values can be changed by a user with the *AUDIT special authority.

QALWUSRDMN: QALWUSRDMN allows you to specify which libraries on the system may contain user domain *USRSPC (user space), *USRIDX (user index), and *USRQ (user queue) user objects.

A change to this system value takes effect immediately. If you specify *ALL, all libraries on the system may contain user domain *USRSPC, *USRIDX, and *USRQ user objects. If you specify a list of library names, applications that currently work with these user domain (*USRxxx) objects may fail if they use objects in libraries not specified by the system value. If a list of libraries is specified, QTEMP must be in the list.

Type	Length	Shipped Value
Character	500	'*ALL'

QAUDCTL: QAUDCTL specifies when object- and user-level auditing is active. This system value activates auditing on the system that is selected by the Change Object Audit (CHGOBJAUD) and Change User Audit (CHGUSRAUD) commands, as well as the QAUDLVL system value. When the QAUDCTL system value is changed to a value other than *NONE, a journal entry is sent to the QAUDJRN security journal to test the auditing journal. If sending this journal entry fails, the change to QAUDCTL is not allowed.

In order to change this system value to a value other than *NONE, the journal QAUDJRN must exist in library QSYS. The journal QAUDJRN cannot be deleted or moved from the QSYS library until the system value is changed to *NONE. A change to this system value takes effect immediately.

When QAUDCTL is changed to a value other than *NONE, a *SHRUPD lock is applied to journal QAUDJRN in library QSYS.

One or more of the following values may be specified. If you specify *NONE, it must be the only specified value:

- *NONE: No auditing of objects (Change Object Audit (CHGOBJAUD) command) or of user actions (Change User Audit (CHGUSRAUD) command, using the AUDLVL keyword) is done on the system. In addition, no auditing controlled by the QAUDLVL system value is done.
- *OBJAUD: Objects that have been selected for audit using the CHGOBJAUD command are audited.
- *AUDLVL: Auditing changes controlled by the QAUDLVL system value and CHGUSRAUD command using the AUDLVL keyword are done.

Type	Length	Shipped Value
Character	50	Note[1]

Note: [1] If you are skipping over a previous release and QAUDLVL is set to something other than *NONE, QAUDCTL will be set to *AUDLVL.

QAUDENDACN: QAUDENDACN specifies the system's action when audit records cannot be sent to the auditing journal because of errors that occur when the journal entry is sent.

The following are the possible values for the QAUDENDACN system value:

- *NOTIFY: Notification of failure to send the journal entry to the security auditing journal is sent to the QSYSOPR message queue and QSYSMSG message queue (if it exists). The action that caused the audit attempt continues. The system value QAUDCTL is set to *NONE to turn auditing off. After the system auditing code turns the QAUDCTL system value off, an hourly notification is sent to the QSYSOPR message queue and QSYSMSG message queue (if it exists) indicating that the system has turned off auditing. The hourly notification stops once auditing is turned on again. The message ID is CPI2283.
- *PWRDWNSYS: The system ends if the attempt to send the audit data to the security audit journal fails. It ends with a B900 3D10 system reference code. When the system is powered-on again, it comes up in the restricted state. Therefore, this IPL requires an attached console. The system value QAUDCTL is set to *NONE to turn auditing off. On the IPL that follows the power down of the system, a user with at least *AUDIT and *ALLOBJ special authority is required to sign on to the system.

The system value QAUDENDACN only applies to auditing entries sent by the operating system.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Character	10	'*NOTIFY'

QAUDFRCLVL: QAUDFRCLVL specifies the number of journal entries written to the security auditing journal before the journal entry data moves to auxiliary storage. This system value also indicates the amount of auditing data that could be lost if the system ends abnormally.

If auditing entries are moved to auxiliary storage frequently, system performance can decrease. This system value is used to modify the QAUDJRN journal to indicate how often the auditing data is moved to auxiliary storage.

The following are the possible values for the QAUDFRCLVL system value:

- *SYS: The system writes the journal entries to auxiliary storage only when the system, based on internal processing, determines the journal entries should be written. Using this option provides the best auditing performance, but it could also cause the most auditing data loss if the system ends abnormally.
- 1-100: The number of auditing journal entries written to the security auditing journal before the auditing data is written to auxiliary storage. Small values decrease system performance.

If a journal entry is written to the security auditing journal for reasons other than an object auditing entry (such as the Send Journal Entry (SNDJRNE) command) and the force journal entry option is taken, all auditing data is written to auxiliary storage along with this journal entry.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Decimal	(5 0)	'*SYS'

QAUDLVL: QAUDLVL is the security auditing level. This system value specifies the actions that are audited on the system. If *NONE is specified, it must be the only value specified. A change to this system value takes effect immediately for all jobs running on the system. Any of the other values may be used alone or in combinations. The values allowed are:

- *NONE: No auditing occurs on the system.
- *AUTFAIL: Authorization failures are audited.
  - All access (sign-on) failures
  - Incorrect password or user ID entered from a device
- *CREATE: The creation of objects is audited.
  - New objects
  - Objects created to replace existing objects
  - Objects created in the QTEMP library are **not** audited
- *DELETE: All object deletions are audited.
  - All delete operations of external objects on the system
  - Objects deleted from QTEMP are **not** audited
- *JOBDTA: The following actions that affect a job are audited:
  - Job start and job stop data
  - Hold, release, change, disconnect, end, end abnormally, PSR (program start request) attached to prestart job entries, change to another user profile
- *OBJMGT: Function of generic objects is audited.
  - Moves of objects
  - Renames of objects
- *OFCSRV: The following OfficeVision/400* tasks are audited:
  - Changing the system distribution directory
  - Opening a mail log for a different user
- *PGMADP: Adopting authority from a program owner is audited.
- *PGMFAIL: Integrity violations (blocked instruction, validation value failure, domain violation) are audited.
- *PRTDTA: The following printing functions are audited:
  - Printing a spooled file.

- Printing with parameter SPOOL(*NO).

- *SAVRST: Save and restore information is audited.
  - When an object is restored
  - When programs that adopt their owner's user profile are restored
  - When job descriptions that contain user names are restored
  - When ownership and authority information changes for objects that are restored
  - When restore of authority for user profiles is done

- *SECURITY: All security related functions are audited.
  - Changes to object authority
  - Create/change/delete/display/restore operations of user profiles
  - Changes to object ownership
  - Changes to programs (CHGPGM) that will now adopt the owner's profile
  - Changes to system values and network attributes
  - Changes to subsystem routing
  - When the QSECOFR password is reset to the shipped value by DST
  - When the DST security officer password is requested to be defaulted

- *SERVICE: Use of the system service tools. The following commands are audited:
  - Dump Object (DMPOBJ), Dump System Object (DMPSYSOBJ), and Dump Document Library Object (DMPDLO)
  - Start Copy Screen (STRCPYSCN)
  - Start, End, Print, and Delete Communications Trace (STRCMNTRC, ENDCMNTRC, PRTCMNTRC, DLTCMNTRC)
  - Print Error Log (PRTERRLOG)
  - Print Internal Data (PRTINTDTA)
  - Start Service Job (STRSRVJOB)
  - Start System Service Tools (STRSST)
  - Trace Internal (TRCINT)

- *SPLFDTA: The following spooled file functions are audited:
  - Create a spooled file
  - Delete a spooled file
  - Display a spooled file
  - Copy a spooled file
  - Get data from a spooled file (QSPGETSP)
  - Hold a spooled file
  - Release a spooled file
  - Change spooled file attributes (CHGSPLFA)

- *SYSMGT: The following system management tasks are audited:
  - Changes for Operational Assistant functions
  - Operations with network files
  - Changes to the system reply list
  - Changes to HFS registration
  - Changes to the DRDA* relational database directory

Type	Length	Shipped Value
Character	160	'*NONE'

QCRTAUT: QCRTAUT is the create authority system value. This value allows the default public authority for create (CRTxxx) commands to be set system-wide. When the *LIBCRTAUT value of the AUT parameter of a create object command is used to set public authority for an object, the CRTAUT value for the library where the object is created determines what public authority is used for the object. If the CRTAUT value of the library is set to *SYSVAL, the value specified in the QCRTAUT system value is used to set the public authority for the object being created.

Changing the QCRTAUT from the shipped value, *CHANGE, to a more restrictive value of *USE or *EXCLUDE limits access to newly-created objects. The owner of the objects, or the security officer, may need to grant additional authority before the object can be used. An example of this is signing on a newly-created device. When the device is created by the CRTDEVDSP command or automatic configuration, the public authority is set to *USE or *EXCLUDE. Because *CHANGE authority to the device description is required to sign on to the device, the sign-on is not allowed.

Changing the QCRTAUT system value to *ALL allows all users of the system, except those given an authority less than *ALL, to completely control the newly-created objects. The users are able to read, change, delete, and manage the security of these objects.

The values allowed are:

- *CHANGE: Allows you to change the contents of an object.
- *ALL: Allows you to read, change, delete, and manage the security of an object.
- *USE: Allows you to create an object, to display the contents of an object, or to refer to the contents of an attached object when a command being requested must access attached objects and their contents.
- *EXCLUDE: Allows no access to an object.

A change to this system value takes effect immediately for newly-created objects. Existing objects are not affected.

Type	Length	Shipped Value
Character	10	'*CHANGE'

QCRTOBJAUD: QCRTOBJAUD specifies the default auditing value used when objects are created into a library. If the CRTOBJAUD value of the library is set to *SYSVAL, the value specified in QCRTOBJAUD system value is used to set the object auditing value for the object being created.

The object auditing value of an object determines if an auditing entry is sent to the system auditing journal QAUDJRN in library QSYS when the object is used or changed. The auditing entry is sent to the auditing journal only if auditing is currently active on the system. To start auditing, system value QAUDCTL must be set to a value other than *NONE.

A change to this system value takes effect immediately. One of the following values is set for objects created into a library.

- *NONE: This value indicates that no auditing entries are sent for an object when it is used or changed.
- *USRPRF: This value indicates that auditing entries are sent for an object when it is used or changed by a user who is currently being audited. If the user who uses or changes this object is not being audited, no auditing entries are sent. To audit a user, you must use the Change User Auditing (CHGUSRAUD) command to change the user profile of the user to be audited.
- *CHANGE: This value indicates that auditing entries are sent for an object when it is changed.
- *ALL: This value indicates that auditing entries are sent for an object when it is used or changed.

Type	Length	Shipped Value
Character	10	'*NONE'

QDSPSGNINF: QDSPSGNINF is the system value for display sign-on information. This logical system value controls whether the user sees an informational display at sign-on that contains the date and time last signed on and the number of not valid sign-on attempts since the last sign-on. The possible values are:

- '0': The sign-on information is not displayed.
- '1': The sign-on information is displayed.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Character	1	'0'

QINACTITV: QINACTITV specifies the inactive job time-out interval in minutes. It specifies when the system takes action on inactive interactive jobs. The system value QINACTMSGQ determines the action the system takes. Local jobs that are currently signed-on to a remote system are excluded. For example, a work station is directly attached to System A, and System A has the QINACTITV system value set on. If pass-through is used to sign on to System B, this work station is not affected by the time-out value set on System A.

QINACTITV must be one of the following values:

- '*NONE': The system does not check for inactive interactive jobs.
- '5-300': The number of minutes a job can be inactive before action is taken.

A change to this system value takes place immediately.

Type	Length	Shipped Value
Character	10	'*NONE'

QINACTMSGQ: QINACTMSGQ is the system value for the inactive message queue. It specifies the action the system takes when an interactive job has been inactive for an interval of time (the time interval is specified by system value QINACTITV). The interactive job can be ended, disconnected, or a message can be sent to the message queue you specify.

QINACTMSGQ is a 20-character list of up to two 10-character values where the first is the message queue name and the second is the library name.

- *DSCJOB: the interactive job is disconnected, as is any secondary or group jobs associated with it.
- *ENDJOB: the interactive job is ended, along with any secondary job and any group jobs associated with it. If there are many inactive jobs in a subsystem that are to be ended at once, the interactive response time of that subsystem may be slowed. To minimize this impact, the system modifies several job attributes for each job to be ended. The job priority is lowered by 10, the time slice is set to 100 milliseconds, and the purge attribute is set to yes.
- '*inactive-message-queue library*': a message indicating the job is inactive is sent to the specified message queue. If the specified message queue does not exist or is damaged, the messages are sent to the QSYSOPR message queue.

  Possible library values are:

  - *LIBL: Use the library list when locating the inactive message queue.
  - '*library name*': The name of the library where the inactive message queue is located.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Character	20	'*ENDJOB'

QLMTDEVSSN: QLMTDEVSSN is the system value for limiting device sessions. It controls whether a user can sign-on at more than one work station. This does not prevent the user from using group jobs or making a system request (pressing the System Request key) at the same work station. The possible values are:

- '0': A user can sign on at more than one device.
- '1': A user cannot sign on at more than one device.

Type	Length	Shipped Value
Character	1	'0'

QLMTSECOFR: QLMTSECOFR is the system value for limiting QSECOFR device access. It controls whether users with *ALLOBJ or *SERVICE special authority need explicit authority to specific work stations. The possible values are:

- '0': A user with *ALLOBJ or *SERVICE special authority can sign-on any device.
- '1': A user with *ALLOBJ or *SERVICE special authority can sign-on only at a device to which they have explicit authority.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Character	1	'1'

QMAXSGNACN: QMAXSGNACN specifies the maximum sign-on attempts action or how the system reacts when the maximum number of consecutive incorrect sign-on attempts (the system value QMAXSIGN) is reached.

A change to this system value takes effect the next time someone attempts to sign on to the system. The possible values are:

- '1': Vary off device if limit is reached.
- '2': Disable user profile if limit is reached.
- '3': Vary off device and disable user profile if limit is reached.

Type	Length	Shipped Value
Character	1	'3'

QMAXSIGN: QMAXSIGN specifies the maximum number of incorrect sign-on attempts allowed. Incorrect sign-on attempts arise from any of the following circumstances:

- A user ID that is not valid.
- A password that is not valid.
- The user profile does not have authority to the device from which the user ID was entered.

Note: A sign-on attempt is not counted as an incorrect attempt if passwords are required and the user profile has a password of *NONE specified. The user receives a message saying that no password is associated with the user profile. It also is not counted as an incorrect attempt if the program or menu names are not valid or if the user ID is not a valid name on an unsecured system, or if the current library specified is not found.

If the QMAXSIGN value maximum is reached, the action specified by the QMAXSGNACN system value is performed. The device is varied off, the profile is disabled, or both actions are performed. A message is sent to the QSYSMSG message queue if it exists; otherwise, it is sent to QSYSOPR. If a profile is disabled, it must be enabled again before a user can sign on. If a device is varied off, it must be varied on again before a user can sign on. If the controlling subsystem is in the restricted state (so that only one device in it can be used) and the device is varied off, the system is ended and control panel lights on the control panel turn on to indicate that you must perform an IPL. The possible values are:

- 1-25: Maximum number of sign-on attempts allowed.
- *NOMAX: No maximum number of sign-on attempts.

When the number of incorrect sign-on attempts is one less than QMAXSIGN, a message is sent to the display to warn the user that if another incorrect attempt is made, the action specified by the QMAXSGNACN system value is performed. If the QMAXSGNACN value is '2,' the message is CPF 1392 *Next not valid sign-on disables user profile.* If the QMAXSGNACN value is '1' or '3,' the message is CPF 1116 *Next not valid sign-on attempt varies off device.* In some environments, it is not desirable to have this message appear. You cannot prevent this message from being sent, but you can change the message text (by using the CHGMSGD command) to the same text as CPF1107 *Password not valid for system.* Refer to CPF1107 for the values entered for the MSG and SECLVL parameters. A change to this system value takes effect the next time someone attempts to sign-on the system.

Type	Length	Shipped Value
Character	6	'15'

QPWDEXPITV: QPWDEXPITV is the system value for the password expiration interval. It controls the number of days passwords are valid by keeping track of the number of days since you changed your password or created a user profile. For example, if the last date your user profile changed was 60 days ago and you change this value to '2,' you will need to change your password the next time you sign on the system. This provides password security by requiring users to change their passwords after a specified number of days. If the password is not changed within the specified number of days, the user cannot sign on until the password is changed. Seven days before the password ends, you are warned at sign-on time, even if you are not displaying sign-on information (the system value QDSPSGNINF). The possible values are:

- '*NOMAX': A password can be used an unlimited number of days.
- '1'-'366': The number of days before the password cannot be used.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Character	6	'*NOMAX'

QPWDLMTAJC: QPWDLMTAJC limits adjacent digits in a password. It specifies whether adjacent digits are allowed in passwords. This makes it difficult to guess passwords by preventing use of dates or social security numbers as passwords. The possible values are:

- '0': Adjacent digits are allowed in passwords.

- '1': Adjacent digits are not allowed in passwords.

A change to this system value takes effect the next time a password is changed.

Note: The password composition system values apply only to password changes made using the CHGPWD command. The CRTUSRPRF and CHGUSRPRF commands ignore password composition system values.

Type	Length	Shipped Value
Character	1	'0'

QPWDLMTCHR: QPWDLMTCHR limits the use of certain characters in a password. This value provides password security by preventing certain characters (vowels, for example) from being used in a password. This makes it difficult to guess passwords by preventing use of common words or names as passwords. The possible values are:

- *NONE: There are no restricted characters.
- restricted-characters: Up to 10 restricted characters can be specified. Valid characters are A through Z, 0 through 9, and special characters such as #, $, _, or @. The specified characters must be enclosed in apostrophes. For example, '0123456789' or 'A_G5$@LU#'.

  Note: '0123456789' cannot be specified if numbers are required in a password (see the system value QPWDRQDDGT).

A change to this system value takes effect the next time a password is changed.

Note: The password composition system values apply only to password changes made using the CHGPWD command. The CRTUSRPRF and CHGUSRPRF commands ignore password composition system values.

Type	Length	Shipped Value
Character	10	'*NONE'

QPWDLMTREP: QPWDLMTREP limits the use of repeating characters in a password. This prevents a user from using the same character more than once in the same password. (For example, AAAA). The possible values are:

- '0': Characters can be used more than once.
- '1': Characters cannot be used more than once.

A change to this system value takes effect the next time a password is changed.

Note: The password composition system values apply only to password changes made using the CHGPWD command. The CRTUSRPRF and CHGUSRPRF commands ignore password composition system values.

Type	Length	Shipped Value
Character	1	'0'

QPWDMAXLEN: QPWDMAXLEN specifies the maximum length of a password. It controls the maximum number of characters in a password.

Note: The maximum password size can not be smaller than the minimum password size specified in the QPWDMINLEN system value. The possible values are:

- 1-10: The maximum number of characters that can be specified for a password.

A change to this system value takes effect the next time a password is changed.

Note: The password composition system values apply only to password changes made using the CHGPWD command. The CRTUSRPRF and CHGUSRPRF commands ignore password composition system values.

Type	Length	Shipped Value
Decimal	(5 0)	10

QPWDMINLEN: QPWDMINLEN specifies the minimum length of a password. It controls the minimum number of characters in a password.

Note: The minimum password size can not be larger than the maximum password size specified in the QPWDMAXLEN system value. The possible values are:

- 1-10: The minimum number of characters that can be specified for a password.

Note: The password composition system values apply only to password changes made using the CHGPWD command. The CRTUSRPRF and CHGUSRPRF commands ignore password composition system values.

Type	Length	Shipped Value
Decimal	(5 0)	1

QPWDPOSDIF: QPWDPOSDIF limits password character positions. It controls the position of characters in a new password. This prevents the user from specifying the same character in a password corresponding to the same position in the previous password. For example, new password ALB2 could not be used if the previous password was ALB1 (the A, L, and B are in the same positions). The possible values are:

- '0': The same characters can be used in a position corresponding to the same position in the previous password.
- '1': The same characters cannot be used in a position corresponding to the same position in the previous password.

Note: The password composition system values apply only to password changes made using the CHGPWD command. The CRTUSRPRF and CHGUSRPRF commands ignore password composition system values.

Type	Length	Shipped Value
Character	1	'0'

QPWDRQDDGT: QPWDRQDDGT specifies whether a digit is required in a new password. This prevents the user from only using alphabetic characters. The possible values are:

- '0': A numeric digit is not required in new passwords.
- '1': A numeric digit is required in new passwords.

Note: '0123456789' cannot be specified for the system value QPWDLMTCHR if numbers are required in a password.

A change to this system value takes effect the next time a password is changed.

Note: The password composition system values apply only to password changes made using the CHGPWD command. The CRTUSRPRF and CHGUSRPRF commands ignore password composition system values.

Type	Length	Shipped Value
Character	1	'0'

QPWDRQDDIF: QPWDRQDDIF controls duplicate passwords. It specifies whether the password must be different than the previous 32 passwords. The possible values are:

- '0': A password can be the same as any previously used password (except the immediately preceding password).
- '1': A password must be different from the previous 32 passwords.

A change to this system value takes effect the next time a password is changed.

Note: The password composition system values apply only to password changes made using the CHGPWD command. The CRTUSRPRF and CHGUSRPRF commands ignore password composition system values.

Type	Length	Shipped Value
Character	1	'0'

QPWDVLDPGM: QPWDVLDPGM is the password validation program. It provides the ability for a user-written program to do additional validation on passwords. The possible values are:

- '*NONE': No validation program is used.
- '*program library*': Specify the name of the validation program and the library (if known) containing the program. For example, 'VALIDPGM' or 'VALIDPGM PGMLIB'.

A change to this system value takes effect the next time a password is changed.

Note: The password composition system values apply only to password changes made using the CHGPWD command. The CRTUSRPRF and CHGUSRPRF commands ignore password composition system values.

Type	Length	Shipped Value
Character	20	'*NONE'

QRMTSIGN: QRMTSIGN is the system value for remote sign-on control. It specifies how the system handles remote sign-on requests. QRMTSIGN can have the following values:

- '*FRCSIGNON': All remote sign-on sessions are required to go through normal sign-on processing.
- '*SAMEPRF': When the source and target user profile names are the same, the sign-on may be bypassed for remote sign-on attempts.
- '*VERIFY': After verifying that the user has access to the system, the system allows the user to bypass the sign-on.
- '*REJECT': No remote sign-on is allowed.
- '*program library*': The user can specify a program and library (or *LIBL) to decide which remote sessions are allowed and which user profiles can be automatically signed on from which locations.

A change to this system value takes effect immediately.

Type	Length	Shipped Value
Character	20	'*FRCSIGNON'

Note: The display station pass-through chapter in the *Remote Work Station Guide* describes QRMTSIGN in greater detail.

QSECURITY: QSECURITY is the system security level indicator.

For unattended IPL, previously requested changes to the security level are made during the IPL.

QSECURITY can be:

- '10': The system does not require a password to sign-on. The user has access to all system resources. Special authorities can be added or removed from a user profile using the CRTUSRPRF or CHGUSRPRF command.
- '20': The system requires a password to sign-on. The user has access to all system resources.
- '30': The system requires a password to sign-on and users must have authority to access objects and system resources.
- '40': The system requires a password to sign-on and users must have authority to access objects and system resources. Programs that try to access objects through interfaces that are not supported will fail.
- '50': The system requires a password to sign on, and users must have authorization to access objects and system resources. The security and integrity of the QTEMP library is enforced. Programs fail if they try to pass unsupported parameter values to supported interfaces, or if they try to access objects through interfaces that are not supported.

  For more information on security level 50, see the *Security Reference* and the *Guide to Enabling C2 Security*.

All user profiles will have special authorities granted or revoked based on their user class when changing from a security level 10 or 20 system to security level 30 or higher.

The QSECURITY system value can be changed at any time but the change to the security level does not take effect until the next IPL as follows:

For attended IPL, the security level required to sign on for the IPL will stay the same as what was in effect before an IPL was performed on the system. Changes are made during the IPL but after the sign on for the IPL.

Note: During an attended IPL, you can also change the QSECURITY system value by selecting 'Define or Change System' on the IPL Options display and the change will take effect when the IPL is done. This change would override any change that was requested before the IPL.

Type	Length	Shipped Value
Character	2	'10'

Network Attributes

Network attributes contain specifications that can be used for networking and communications. Network attributes are not objects and cannot be passed as parameter values like CL variables.

The system is shipped with certain network attributes. Most of these are important only if your system is part of a communications network. The system name appears on many OS/400 displays, including the sign-on display.

The network attributes shipped with the system are shown in the following figure. The manuals listed in the figure have more information about each network attribute.

Figure 2-9 (Page 1 of 2). Network Attributes Shipped with the System

Description	Shipped Value	For More Information
System name	SYSNAME(based-on-machine-serial-number)	See the *APPC Programmer's Guide*.
Local network ID to be assigned to the system	LCLNETID(APPN)	See the *APPC Programmer's Guide*.
Local control point name for the system	LCLCPNAME(based- on-machine- serial-number)	See the *APPC Programmer's Guide*.
Default local location name for the system	LCLLOCNAME(based-on-machine-serial- number)	See the *APPC Programmer's Guide*.
Default mode name for the system	DFTMODE(BLANK)	See the *APPC Programmer's Guide*.
APPN node type	NODETYPE(*ENDNODE)	See the *APPC Programmer's Guide*.
Data compression	DTACPR(*NONE)	See the *APPC Programmer's Guide*
Intermediate data compression	DTACPRINM(*NONE)	See the *APPC Programmer's Guide*
Maximum number of intermediate sessions	MAXINTSSN(200)	See the *APPC Programmer's Guide*.
APPN route addition resistance	RAR(128)	See the *APPC Programmer's Guide*.
APPN network node servers	NETSERVER(*NONE)	See the *APPC Programmer's Guide*.
Alert status	ALRSTS(*OFF)	See the *Alerts and DSNX Guide*.
Specifies which alerts are logged	ALRLOGSTS(*NONE)	See the *Alerts and DSNX Guide*.
Specifies if the system is an alert primary focal point	ALRPRIFP(*NO)	See the *Alerts and DSNX Guide*.
Specifies if the system is an alert default focal point	ALRDFTFP(*NO)	See the *Alerts and DSNX Guide*.
Identifies the system that provides backup alert focal point services	ALRBCKFP(*NONE)	See the *Alerts and DSNX Guide*.
Identifies the system that you requested as the alert focal point	ALRRQSFP(*NONE)	See the *Alerts and DSNX Guide*.
Name of the controller through which alerts are sent on the SSCP-PU session	ALRCTLD(*NONE)	See the *Alerts and DSNX Guide*.
Number of alerts that are accumulated before they are sent	ALRHLDCNT(0)	See the *Alerts and DSNX Guide*.
The filter object used by the alert manager when processing alerts	ALRFTR(*NONE)	See the *Alerts and DSNX Guide*.
Default message queue used in object distribution	MSGQ(QSYS/QSYSOPR)	See the *Distribution Services Network Guide*.

Figure 2-9 (Page 2 of 2). Network Attributes Shipped with the System

Description	Shipped Value	For More Information
Default output queue used in object distribution	OUTQ(QGPL/QPRINT)	See the *Distribution Services Network Guide*.
Default action taken for job streams received by the system (used in object distribution)	JOBACN(*FILE)	See the *Distribution Services Network Guide*.
Maximum number of systems a distribution can pass through on its path to a destination on a SNADS network	MAXHOP(255)	See the *Distribution Services Network Guide*.
The action to be taken for DDM requests from other systems	DDMACC(*OBJAUT)	See the *DDM Guide*.
Specifies how PC Support requests are handled	PCSACC(*OBJAUT)	See the *PC Support/400 Technical Reference for DOS and OS/2*.
ISDN network type	DFTNETTYPE(blank)	See the *APPC Programmer's Guide*.
ISDN connection list	DFTCNNLST(QDCCNNLANY)	See the *APPC Programmer's Guide*.

How To's

The following section tells you with how to perform some of the common tasks involving system values and network attributes.

Displaying or Changing a System Value

Two of the more common tasks you will perform on system values are displaying and changing them. AS/400 work management provides you with the Work With System Value (WRKSYSVAL) command to make these tasks easy. The WRKSYSVAL command gives you the list of all system values, or allows you to subset that list, so you can select the system value you want to display or change. The WRKSYSVAL command is an alternative to using either the Display System Value (DSPSYSVAL) or Change System Value (CHGSYSVAL) command, and should be used when you are unsure of the name of the system value you want to affect, or you want to see the current value before making a change. If a system value is currently being changed, the system may prevent you from displaying, retrieving, or extracting that specific system value. Both DSPSYSVAL and CHGSYSVAL require you to know the name of the system value you want to display or change.

After entering the WRKSYSVAL command a display appears. On the display you can request a list of all the system values, or you can subset that list by entering a type in the Subset by Type field. The following display illustrates how to request a list of all of the system values:

```
                          Work with System Values
                                                              System:   SYS01
Position to  . . . . . .                 Starting characters of system value
Subset by Type . . . . .  *ALL           F4 for list

Type options, press Enter.
  2=Change   5=Display

       System
Option Value       Type     Text
  _    QTIME       *DATTIM  Time of day
  _    QTIMSEP     *EDT     Time separator
  _    QTOTJOB     *ALC     Initial total number of jobs
  _    QTSEPOOL    *STG     Time slice end pool
  _    QUPSDLYTIM  *SYSCTL  Uninterruptible power supply delay time
  _    QUPSMSGQ    *SYSCTL  Uninterruptible power supply message queue
  _    QUSRLIBL    *LIBL    User part of the library list

                                                                       Bottom
Command
===>
F3=Exit   F4=Prompt   F5=Refresh   F9=Retrieve   F11=Display names only
F12=Cancel
```

You can use the Display System Value (DSPSYSVAL) command to display a system value whose name you know.

To change a system value whose name you know, you can use the Change System Value (CHGSYSVAL) command. You can use the CHGSYSVAL command to change system values within CL programs.

If you are using the CHGSYSVAL command, system values must be enclosed in apostrophes under three conditions:

- If the system value specified is a character string with embedded blanks

- If numeric values or special characters are specified for character type system values
- If the system value is a date or time value

The system values (QACGLVL, QALWUSRDMN, QAUDCTL, QAUDLVL, QCHRID, QCMNRCYLMT, QSYSLIBL, QUPSDLYTIM, QUSRLIBL, and QIPLDATTIM) may be lists. To separate items in the list, use blanks and enclose the entire list in apostrophes. If there is only one item in the list, you do not need apostrophes. The QUPSDLYTIM system value is a list of two items but only the first item is used when a Change System Value (CHGSYSVAL) command is run. The second item, if specified, is ignored.

The system values (QATNPGM, QCTLSBSD, QIGCCDEFNT, QINACTMSGQ, QPRBFTR, QPWDVLDPGM, QRMTSIGN, QSRTSEQ, QSTRUPPGM, and QUPSMSGQ) may be qualified with a library name. If the system values are qualified, use blanks to separate the name and library, enclose the value in apostrophes (for example, 'SBSD LIB'), and specify the object first followed by the library ('object library'). Apostrophes are necessary only when the library name or *LIBL is specified with the object name. If the library name is not specified or *LIBL is specified for the library, the library list is used to locate the object, and the library where the object is found is stored in the system value.

Note: When object names are specified for system values, the lowercase letters in the names are always changed to uppercase even when they are in apostrophes. This means that you should not use lowercase letters in the names of objects or libraries that you may want to specify on any of the system values.

Example Using the Change System Value Command: The following CHGSYSVAL command changes the value of the system value QUSRLIBL:

```
CHGSYSVAL SYSVAL(QUSRLIBL)
          VALUE('DSTPRODLB QGPL QTEMP')
```

This system value defines the system default for the user part of the initial library list. The initial library list for new jobs is redefined and DSTPRODLB is placed before QGPL and QTEMP. The change takes effect immediately for jobs started after the CHGSYSVAL command is run (but not for jobs that are already active).

How the value is entered on the VALUE parameter is dependent on the type of system value: character or numeric. The following rules for specifying the VALUE parameter apply to system values.

- If a character string contains characters other than alphanumeric characters, a period, or a comma, it must be enclosed in apostrophes.
- If a list of values is specified as the new value, the list must be enclosed in apostrophes and separated by blanks.
- If a numeric value or special character is specified for a character type system value, it must be enclosed in apostrophes.
- Date-and-time values must be enclosed in apostrophes and cannot contain separators.
- All qualified names in system values should be specified as 'object library'.

All numeric type system values must be specified as whole numbers greater than or equal to 0 and not exceeding 32767. If a numeric system value is specified by using a CL variable, the variable must be decimal. A numeric value is not enclosed in apostrophes unless it is specified for a character type system value. Each system value has its own set of limitations on values. If a CL variable is used to specify the value for a character type system value, the CL variable must be character type and may be of any length. Values are padded with blanks or truncated, as necessary.

Not all system value changes are used by the system immediately when a system value is changed. For some values, such as QCTLSBSD, the change takes effect the next time an IPL is done.

Retrieving a System Value

Use the Retrieve System Value (RTVSYSVAL) command to retrieve a system value.

Displaying Network Attributes

Use the Display Network Attributes (DSPNETA) command to display a network attribute.

Changing Network Attributes

Use the Change Network Attribute (CHGNETA) command to change network attributes.

Retrieving Network Attributes

Use the Retrieve Network Attributes (RTVNETA) command to retrieve network attributes. The RTVNETA command description in the *CL Reference* contains a detailed description of the network attributes.

Chapter 3. Subsystems

A subsystem is a single, predefined operating environment through which the system coordinates the work flow and resource use. The system can contain several subsystems, all operating independently of each other. Subsystems manage resources. The run-time characteristics of a subsystem are defined in an object called a subsystem description.

This chapter describes the parts that make up a subsystem description, such as subsystem attributes, work entries, routing entries, pools, and activity levels. "How To's" on page 3-16 tells you how to perform some common tasks involving subsystems.

Subsystem Descriptions

A subsystem description defines how, where, and how much work enters a subsystem, and which resources the subsystem uses to perform the work. An active subsystem takes on the simple name of the subsystem description. You can use the subsystem description supplied with your system (with or without making changes to it), or you can create your own subsystem descriptions.

A subsystem description consists of three parts:

- Subsystem attributes (overall subsystem characteristics)
- Work entries (sources of work)
- Routing entries

Subsystem Attributes

Subsystem attributes provide the overall characteristics of the subsystem. The following is a list of attributes:

- Operational attributes including maximum number of jobs that can be active in the subsystem at the same time, the sign-on display file (to use for interactive work), and the system library list entry
- Pools of main storage used by the subsystem
- Authority to subsystem description
- Text description of the subsystem description

You can use the CHGSBSD command to change the storage pool size and storage pool activity level for subsystem descriptions (or use the WRKSYSSTS command for active subsystems). See "How To's" on page 3-16 for more information on how to change pool sizes and activity levels.

Work Entries (Sources of Work)

Work entries specify the sources from which work can enter the subsystem. The following is a list of work entries (see "Work Entries" on page 3-7 for a complete description of each type of work entry):

- Autostart job entry
- Work station entry
- Job queue entry
- Communications entry
- Prestart job entry

You can work with existing work entries while the subsystem is active:

- Use the Change Job Queue Entry (CHGJOBQE) command to change an existing job queue entry.
- Use the Change Prestart Job Entry (CHGPJE) commands to work with a prestart job entry.
- Use Add Autostart Job Entry (ADDAJE), Change Autostart Job Entry (CHGAJE), and Remove Autostart Job Entry (RMVAJE) commands to work with autostart job entries.

All other changes to the subsystem description can only be made when the subsystem is inactive, including adding or removing storage pools. (An active subsystem is one that has been started, for example, with the Start Subsystem (STRSBS) command; an inactive subsystem is one that has been ended, for example, with the End Subsystem (ENDSBS) command, or has not been started.)

Double-Byte Character Set Considerations:
IBM-supplied subsystem descriptions are changed for controlling double-byte character set devices. You do not need to change any subsystem descriptions.

Routing Entries, Routing Steps, and Routing Data

Routing entries in a subsystem description specify the program to be called to control a routing step for a job running in the subsystem, which pool the job will use, and from which class to get the run-time attributes. Routing data identifies a routing entry for the job to use. The **routing step** is the processing performed as a result of calling the program specified in a routing entry. The processing done for a job running in a subsystem consists of one or more consecutive routing steps. Usually, a job has only one routing step.

Pools and Activity Levels

On the AS/400 system, all main storage can be divided into logical allocations called storage pools.

There are 2 basic types of storage pools on the system, shared pools and private pools.

Shared Pools

A **shared storage pool** is a pool in which multiple subsystems can run jobs. Using shared storage pools allows the system to distribute the storage requirements of interactive users across multiple subsystems, still allowing their jobs to run in the same storage pool.

There are 14 shared storage pools defined on the system, 13 of which you can specify for use when creating subsystem descriptions (the machine pool is reserved for system use). Shared storage pools include the following:

- *MACHINE is the machine storage pool that is used for highly shared machine and OS/400 licensed programs. The size for this storage pool is specified in the system value QMCHPOOL. No user jobs run in this storage pool. On the Work with System Status display (WRKSYSSTS), the machine storage pool appears as system pool identifier 1.
- *BASE is the base storage pool that contains all unassigned main storage on the system; that is, all storage that is not required by the machine storage pool or by another pool. The system value QBASPOOL specifies the minimum size of the base storage pool. The activity level for this storage pool is specified in the system value QBASACTLVL. The base storage pool is used for batch work and miscellaneous system functions. (On the Work with System Status display (WRKSYSSTS), the base storage pool appears as system pool identifier 2.)
- *INTERACT is the interactive storage pool used for interactive jobs.
- *SPOOL is the storage pool used for spool writers.
- *SHRPOOL1 through *SHRPOOL10 are storage pools that you can use for your own use.

To change the size or activity level of shared pools, you need authorization to the Change Shared Storage Pool (CHGSHRPOOL) command. On the Work with System Status (WRKSYSSTS) display, you cannot change the activity level of the *MACHINE pool or the size of the *BASE pool.

Changes to shared pools take effect immediately if the shared pool is active and sufficient storage is available.

Private Pools

A **private storage pool** is a pool in which a single subsystem can run jobs. A private pool does not have to be large enough to contain your programs. The machine manages the transfer of data (and programs) into the private storage pool if necessary. If data is already in main storage, it can be referred to independently of the storage pool it is in. However, if needed data does not exist in any storage pool, it is brought into the same storage pool for the job that referred to it (this is known as a page fault). As data is transferred into a storage pool, other data is displaced and, if changed, is automatically recorded in auxiliary storage (this is called paging). The storage pool size should be large enough to keep data transfers (paging) at a reasonable level as the rate affects performance. For information about changing the size of a pool, see Chapter 13, "Performance Tuning."

When you create or change a subsystem description, you can define one or more pools, which are labeled 1, 2, 3, and so on. These are the designations of the subsystem pools, and do

not correspond to the pool numbers shown on the system status display. The Work with Subsystems display (WRKSBS) relates the subsystem pool identifiers and the column headings to the system pool identifiers. See "Pool Numbering" on page 3-4 for information about how pools are numbered.

A system job, called a subsystem monitor job, controls all the activity in a subsystem. Storage for the subsystem monitor comes from pool 1 of the subsystem, but does not count toward an activity level. See Chapter 12, "System Jobs," for more information about the subsystem monitor.

The shipped default causes the storage for the subsystem monitor to come from the base storage pool. If you want storage for the subsystem monitor to come from a separate pool, specify the CRTSBSD or CHGSBSD command with a specific pool size and activity level such as:

```
POOLS((1 300 2))
```

For more information on the machine storage pool and the base storage pool, see Chapter 13, "Performance Tuning."

In addition to the machine storage pool and the base storage pool, you can create user-defined storage pools. User-defined storage pools can be used by IBM-supplied subsystems or user-defined subsystems. To create a user-defined storage pool, use the POOLS parameter on the Create Subsystem Description (CRTSBSD) or Change Subsystem Description (CHGSBSD) commands.

Multiple Pools in a Subsystem: Multiple pools help you control the jobs' competition for system resources. The advantages of having multiple pools in a subsystem are that you can separate the amount of work done and the response time for these jobs. For example, during the day you may want interactive jobs running with good response time. For better efficiency, you can make the interactive pool larger. At night you may be running many batch jobs, so you make the batch pool larger.

Note: It is possible to divide the work too much, giving you more to tune and manage on the system.

Allocating Pools

When you start a subsystem, the system attempts to allocate the user-defined storage pools defined in the subsystem description of the started subsystem.

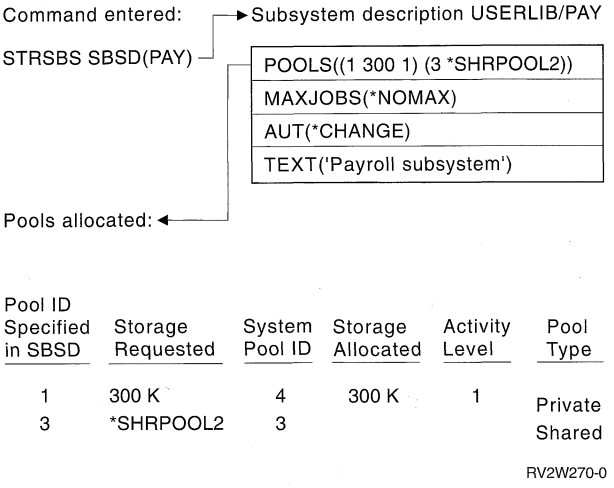


Figure 3-1. Starting a Subsystem

If the system cannot allocate all the requested storage, it allocates as much storage as is available and allocates all the other as storage becomes available. For example, if, in the previous example, there was only 700KB available, 300KB would have been allocated to the first storage pool and 400KB would have been allocated to the second storage pool.

The storage pools that you define decrease the size of the base storage pool. The system does not allocate storage to a user pool if the amount of storage available to the base storage pool would be less than the minimum base pool size specified by the system value QBASPOOL.

Activity Levels

The **activity level** of a storage pool is the number of jobs that can run at the same time in a storage pool. The machine manages the control of this level. Often during processing in a job, a program waits for a system resource or a response from a work station user. During such waits, a job gives up its use of the storage pool in order that another job that is ready to be processed can take its place.

More than one job can be active at the same time in a storage pool because the processing for a job can be briefly interrupted while needed data is gotten from auxiliary storage. During this delay, which is usually short, another job can run. Using the activity level, the machine can process a large number of jobs in a storage pool and, at the same time, hold the level of contention to the limit you specify.

Once the maximum activity level for a storage pool has been reached, additional jobs needing the storage pool are automatically put in a queue to wait for available main storage (placed in an ineligible state, that is shown on the Work with Active Jobs Display). As soon as a job gives up its use of the storage pool, the jobs in the queue become eligible to run by their priority. For example, if a running job is waiting for a response from a work station, it gives up its activity level and the activity level is no longer at its maximum.

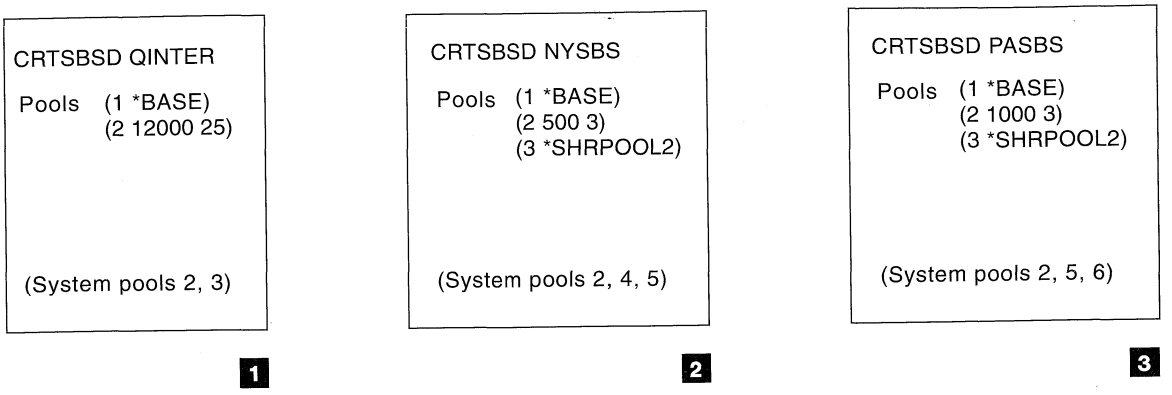
Defining storage pools and activity levels correctly is generally dependent on the number of jobs a subsystem must support at the same time and the characteristics and use of the programs that perform functions in those jobs. See Chapter 13, "Performance Tuning," for a more detailed description of how to set appropriate activity levels.

The storage pool that user jobs get their storage from is always the same pool that limits their activity level. System jobs (such as SCPF, QSYSARB, and QLUS) get their storage from the base pool, but use the machine pool activity level. Subsystem monitors get their storage from the first subsystem description pool but not the activity level. This allows a subsystem monitor to always be able to run regardless of the activity level setting.

Pool Numbering

Pool numbering can be confusing because two sets of numbers are used for numbering storage pools. Subsystems use one set of numbers to refer to the pools they use, and the system uses another set of numbers to keep track of all the pools. Figure 3-2 on page 3-5 illustrates how pools are numbered.

Subsystems



After QINTER starts, the following pools are allocated:

System Pool Number	Description	QINTER
1	*Machine pool	
2	*BASE pool	1
3	QINTER private pool	2

After NYSBS starts, the following pools are allocated:

System Pool Number	Description	QINTER	NYSBS
1	*Machine pool		
2	*BASE pool	1	1
3	QINTER private pool	2	
4	*SHRPOOL2 shared pool		3
5	NYSBS private pool		2

After PASBS starts, the following pools are allocated:

System Pool Number	Description	QINTER	NYSBS	PASBS
1	*Machine pool			
2	*BASE pool	1	1	1
3	QINTER private pool	2		
4	*SHRPOOL2 shared pool		3	3
5	NYSBS private pool		2	
6	PASBS private pool			2

RV2W266-1

Figure 3-2. Pool Numbering

You can also use the Work with Subsystem (WRKSBS) command to find information about system pool numbers for a given subsystem. Type the WRKSBS command on a command line and press the Enter key to obtain the following display:

```
                              Work with Subsystems
                                                                 System:   RCH38342
Type options, press Enter.
  4=End subsystem   5=Display subsystem description
  8=Work with subsystem jobs

                    Total     ----------Subsystem Pools----------
Opt  Subsystem   Storage (K)   1   2   3   4   5   6   7   8   9  10
     NYSBS            5000     2   4   5
     PASBS            1000     2   6   5
     QINTER          12000     2   3


                                                                          Bottom
Parameters or command
===>
F3=Exit   F5=Refresh   F11=Display System data   F12=Cancel
F14=Work with System status
```

Note that the subsystem names are listed alphabetically in this display. Also, the numbers under the *Total Storage* column indicate the total storage allocated to the subsystem private pools. Storage from shared pools is not included in this total.

Controlling Levels of Activity

The following specify levels of control over the starting of jobs:

- A subsystem. The MAXJOBS parameter on the CRTSBSD and CHGSBSD commands is used to specify how many jobs can be active at the same time in a subsystem. For an active subsystem, the sum of all jobs that are active at the same time started through work entries in the subsystem cannot exceed the MAXJOBS parameter value. This excludes autostart jobs, which may temporarily cause the limit to be exceeded when the subsystem is started.
- A job queue entry. The MAXACT parameter on the ADDJOBQE and CHGJOBQE commands is used to specify how many batch jobs from a job queue can be active at the same time in the subsystem. (A MAXACT of 1 for a job queue forces jobs to be selected serially by job priority from a job queue.) The MAXPTYn parameter is used to specify how many jobs can be active for a specified job priority.
- A work station entry. If the WRKSTNTYPE parameter is specified, the MAXACT parameter on the ADDWSE and CHGWSE commands is used to specify how many interactive jobs can be active at the same time in the subsystem for that entry.
- A communications entry. The MAXACT parameter on the ADDCMNE and CHGCMNE commands is used to specify how many communications batch jobs can be active at the same time for that entry.
- A routing entry. The MAXACT parameter on the ADDRTGE and CHGRTGE commands is used to specify how many jobs can be active at the same time using a given routing entry.
- A prestart job entry. The MAXJOBS parameter on the ADDPJE and CHGPJE commands is used to specify how many prestart jobs can be active at the same time for that entry.

For the MAXJOBS and MAXACT parameters, active jobs refer to jobs that have started running but have not completed running. Active jobs do not include jobs that are on a job queue waiting to be started, or jobs that have completed processing but are waiting for a spooled file to be printed.

The following specify levels of control over use of the processing unit for jobs which are already started:

- The system. The system value QMAXACTLVL (see Chapter 2, "System Values and Network Attributes") is used to specify how many jobs can share main storage and processor resources at the same time. All active jobs (including system jobs) in all storage pools are controlled by QMAXACTLVL.
- The base storage pool. The system value QBASACTLVL (see Chapter 2, "System Values and Network Attributes") is used to specify how many jobs can share the base storage pool at the same time and compete for use of the processing unit.
- Shared pools. Use the Work with Shared Pools (WRKSHRPOOL) command to specify the activity level for shared pools.
- Private storage pools. Use the POOLS parameter on the CRTSBSD and CHGSBSD commands to specify the activity level for user-defined main storage pools.

Activity level refers to jobs sharing the processing unit at the same time. These jobs are all active jobs. When more jobs are started than can run at the same time because of the activity level controls, the excess jobs have to wait to use the processing unit (normally this wait is very short).

For the storage pool activity level, the number of jobs running (or active jobs) refers to the number of jobs that have an activity level; that is, the jobs are actually running or they may be waiting for a disk I/O operation. In this sense, active jobs do not refer to jobs that are waiting for input, for a message, for a device to be allocated, or for a file to be opened. Active jobs do not refer to jobs that are ineligible (jobs that are ready to run but the storage pool activity level is at its maximum). See Chapter 13, "Performance Tuning," for a description of the states of a job in relation to the storage pool activity level.

The number of batch jobs that are allowed to start and the activity level specified for storage pools are the most useful. The storage pool activity level lets you limit the amount of running contention in the various storage pools in your subsystems. For the other activity limits (the number of jobs you allow to be started), you typically specify a value of *NOMAX unless special circumstances require that an actual limit be imposed.

The following examples show the relationship of some of the activity controls. The system activity level is assumed to be 100.

Example 1: Two subsystems, SBSA and SBSB, use the base storage pool to run jobs. SBSA currently has two jobs running in this storage pool and SBSB has one. A job queue entry in the subsystem description for SBSB specifies that any number of jobs can be started. The activity level of the base storage pool is 3. Therefore, only three jobs in the base storage pool can compete for the processing unit at a time. However, all the jobs are started.

Example 2: One autostart job, two work station jobs, and one batch job–four jobs in all–are in subsystem SBSC. The MAXACT for SBSC is specified as 4. No matter what is specified for the MAXACT of the work entries, no other jobs can be started until at least one job completes running.

Example 3: Subsystem SBSE is a batch subsystem for which 1 is specified for MAXACT. Although the job queue entry does not specify MAXACT, the limit is one job because 1 is specified for MAXACT for the subsystem. Therefore, jobs are processed in job priority one at a time off the job queue.

Work Entries

Work entries identify the sources where jobs can enter a subsystem. The following sections provide definitions of each type of work entry and related parameters. See "How To's" on page 3-16 for information about adding and changing work entries in a subsystem description. Once a subsystem is active, only the MAXACT can be changed on existing job queue entries.

Autostart Job Entry

The job is automatically started each time the subsystem is started. You can specify the following items in an autostart job entry. Parameter names are given in parentheses.

- Job name (JOB)
- Job description name (JOBD)

To specify which program or command to run, use the request data (RQSDTA) parameter on the job description. See Chapter 8, "Autostart Jobs," for more information.

Job Queue Entry

Jobs to be processed are taken from the specified job queue. You can specify the following items in a job queue entry. Parameter names are given in parentheses.

- Job queue name (JOBQ)
- Maximum number of jobs that can be active at the same time from the job queue (MAXACT)
- Order in which the subsystem selects job queues from which jobs can be started (SEQNBR)
- Maximum number of jobs that can be active at the same time for a specified job queue priority (MAXPTYn)

See Chapter 7, "Batch Jobs," for more information.

Work Station Entry

The job is started when a work station user signs on or when a work station user transfers an interactive job from another subsystem. You can specify the following items in a work station entry. Parameter names are given in parentheses.

- Work station name or type (WRKSTN or WRKSTNTYPE)
- Job description name (JOBD) or job description name in the user profile
- Maximum number of jobs that can be active at the same time through the entry (MAXACT)
- When the work stations are to be allocated, either when the subsystem is started or when an interactive job enters the subsystem through the Transfer Job (TFRJOB) command (AT)

See Chapter 5, "Interactive Jobs," for more information.

Generic Work Station Names and Types:

You can use generic names for work station name entries on the work station entry commands, such as Add Work Station Entry (ADDWSE), Change Work Station Entry (CHGWSE), and Remove Work Station Entry (RMVWSE). In addition, special values are allowed for work station type entries. These entries provide the following:

- Allow more control over the allocation of devices to a specific subsystem
- Allow easier addition of work stations to an already active subsystem
- Allow all of the displays on the system to come up under one subsystem without adding all of the possible device types to the subsystem

The support for generic work station names and types on the work station entry commands include the following:

- Generic capability for the work station name function

```
WRKSTN(-name-)

WRKSTN(LOCAL10)

WRKSTN(LOCAL*)
```

- Special generic values for the work station type function

```
                |- type number---|
WRKSTNTYPE ( |                  | )
                |- *ALL ---------|
                |- *NONASCII-----|
```

The order in which work station entries are processed (allocated) for an individual subsystem is as follows:

1. Specific work station name
2. Generic work station name
3. Special work station type (*CONS)
4. Specific work station type (5250, 3279, ...)
5. Special work station type (*ASCII and *NONASCII)
6. Special work station type (*ALL)

When a subsystem contains generic entries which match the device name, the more specific generic name does not have priority.

```
SBSD1                                SBSD2
   WRKSTN(DSP*)                        WRKSTN(D*)
```

The following are examples of how the generic function may be used:

- Local and remote work stations may be divided into separate subsystems by giving them standard names, such as LOCAL10, LOCAL20, RMT10, RMT20, and so on. Generic work station entries, such as LOCAL* and RMT*, may then be added to different subsystem descriptions to separate the local and remote work stations.
- The *ALL or generic entries such as DSP* make it less likely that the subsystem has to be ended when adding a new device to the system. All work stations meeting the generic qualification are dynamically allocated by the subsystem when they are created.
- The *ALL work station type allows the subsystem to allocate all of the valid work stations on the system.

Communications Entry

The job is started when the subsystem receives a program start request from a remote system. You can specify the following items in a communications entry. Parameter names are given in parentheses.

- Device description name or type (DEV)

- Remote location name (RMTLOCNAME)
- Job description name (JOBD), or job description name in the user profile
- Maximum number of jobs that can be active through this communications entry (MAXACT)
- Default user (DFTUSR)
- Mode name (MODE)

See Chapter 9, "Communications Jobs," for more information about communications jobs.

Prestart Job Entry

The jobs are started on a local system before a remote system sends a program start request. You can specify the following items in a prestart job entry. Parameter names are given in parentheses.

- Qualified name of the subsystem description to which the prestart job is added (SBSD)
- Qualified name of the program that the prestart job runs (PGM)
- User profile (USER)
- Simple name of the prestart job (JOB)
- Qualified name of the job description used for the prestart job (JOBD)
- Whether the prestart jobs start at subsystem startup (STRJOBS)
- Initial number of prestart jobs (INLJOBS)
- When additional prestart jobs should start (THRESHOLD)
- Additional number of prestart jobs that should start when the number of prestart jobs drops below the THRESHOLD parameter (ADLJOBS)
- Maximum number of prestart jobs that can be active at the same time (MAXJOBS)
- Maximum number of program start requests that can be handled by each prestart job in the pool before the job is ended (MAXUSE)
- Whether program start requests wait for a prestart job to become available or are rejected if a prestart job is not immediately available when the program start request is received (WAIT)
- Subsystem pool identifier (POOLID)
- Qualified names of the classes and how many prestart jobs run using each class (CLS)

See Chapter 10, "Prestart Jobs," for more information about prestart jobs.

Routing Entries and Routing Data

A routing entry in a subsystem description specifies the program that is called to control a routing step that runs in the subsystem. You use the Add Routing Entry (ADDRTGE) command to add a routing entry. A routing entry contains the following (parameter names are given in parentheses):

- A routing entry sequence number (SEQNBR)
- A routing data comparison value and starting position for the comparison (CMPVAL)
- The name of the program called (PGM) or *RTGDTA, which indicates that the program name is to be taken from the routing data
- The name of the class used for the routing step (CLS)
- The maximum number of routing steps that can be active at the same time for the entry (MAXACT)
- The identifier of the subsystem storage pool in which the job is to run (POOLID)

For batch jobs, routing data comes from the job description or from the routing data (RTGDTA) parameter on the SBMJOB or BCHJOB command. For interactive jobs, routing data comes from the job description.

For transferred or rerouted jobs (either batch or interactive), routing data comes from the TFRJOB, TFRBCHJOB, or Reroute Job (RRTJOB) command. For a communications job, the routing data is created by the subsystem based on information received in the program start request. Routing data is not used for a prestart job. For a prestart job, the name of the program to be started is specified on the prestart job entry.

A routing entry contains values that control the selection of which routing entry is to start the routing step: sequence number, comparison value, and position at which the comparison is to start. Three other values define the running environment for the routing step: program name,

class name, and storage pool identifier. For a prestart job, the class name and storage pool identifier are specified on the prestart job entry.

The sequence number defines the order in which the routing entries are scanned and can be used as the identifier of the routing entry. The comparison value specifies data that is compared with routing data to determine which routing entry to use. (The routing entry also specifies the starting position for the comparison.) The routing data is compared with the comparison value of each routing entry in sequence number order until a match is found.

When a routing entry is found with a comparison value that matches the routing data, a routing step is started and the program specified in the routing entry is called. The run-time attributes in the class associated with the routing entry are used for the routing step, and the routing step runs in the storage pool specified in the routing entry.

For communications considerations for jobs started by program start requests, see the *ICF Programmer's Guide*.

When you add a routing entry to a subsystem description, you assign a sequence number to the entry. This sequence number tells the subsystem the order in which routing entries are to be searched for a routing data match. For example, you have a subsystem description containing the following five routing entries:

Sequence Number	Comparison Value
10	'ABC'
20	'AB'
30	'A'
40	'E'
50	'D'

The routing entries are searched in sequence number order. If the routing data is 'A', the search ends with routing entry 30. If the routing data is 'AB', the search ends with routing entry 20. If the routing data is 'ABC', the search ends with routing entry 10. Because routing data can be longer than the comparison value of the routing entry, the comparison (which is done in left-to-right order) stops when it reaches the end of the comparison value. Therefore, if the routing data is 'ABCD', the search ends with routing entry 10.

When you define routing entries, they must be ordered from the most specific to the most general. The following example shows a correct and incorrect way to define routing entries:

Correct		Incorrect	
Sequence Number	Comparison Value	Sequence Number	Comparison Value
10	'ABC'	10	'ABC'
20	'AB'	20	'ABCD'
30	'A'		
40	'E'		
9999	*ANY		

In the incorrect example, it is no longer possible to match routing entry 20 because any routing data that matches the comparison value for routing entry 20 matches the routing entry 10 first. When a routing entry is changed or added to a subsystem description with a comparison value that causes this situation, the system sends a diagnostic message identifying the situation.

When you add routing entries to a subsystem description, you should order them so that the entries likely to be compared most often are first. This reduces the search time.

You can specify a comparison value of *ANY on the highest numbered routing entry. *ANY means that a match is forced regardless of the routing data. Only one routing entry can contain the comparison value of *ANY, and it must be the last (highest sequence number) entry in the subsystem description. See Appendix B, "IBM-Supplied Object Contents," for an example.

The program named in the routing entry is given control when the routing step for the job is started. Parameters to control the run-time environment (priority, time slice, and so on) of the routing step for the job are taken from the class specified in the routing entry. (See "Run-Time Attributes (Class Object)" on page 4-6.)

Allocating Work Station Devices

When a subsystem is started, it attempts to allocate all work station devices in its subsystem description. The following situations may occur during the time the subsystem starts:

- If the device is not varied on, the subsystem cannot allocate it. The system arbiter

(QSYSARB) holds a lock on all varied-off devices.

- If the device is varied on and has not been allocated by any other subsystem, the subsystem can allocate it and display the Sign On display.
- If the device is varied on and has been allocated by another subsystem and is at the Sign On display (the Sign On display was displayed before the second subsystem was started), a second subsystem can allocate the device from the first subsystem and display the Sign On display.

If more than one subsystem tries to allocate the same work station (as specified in the work station entries) and the work station is varied off, which subsystem that gets the work station when it is varied on cannot be predicted. Similarly, if a work station entry specifies a work station type instead of a work station name, a subsystem may get all, some, or none of the work stations of that type. To avoid such a situation, you can set up the work station entries for the subsystems so that multiple subsystems are not using the same work stations.

When a user signs on to a work station, his job runs in the subsystem that had the sign-on display shown on the work station (the subsystem is identified in the sign-on display). The following situations may occur after the user has signed on:

- If a second subsystem is started and it tries to allocate the work station on which the user signed on, the second subsystem cannot allocate it. The user's job continues to run in the first subsystem.
- If the user selects option 1 (Display sign-on for alternative job) on the System Request menu or issues the Transfer to Secondary Job (TFRSECJOB) command, the new job runs in the same subsystem as the original job.
- When the user signs off, the work station remains allocated to the subsystem used when the user signed on, unless the user transferred into the subsystem using the Transfer Job (TFRJOB) command, and specified AT(*ENTER) for the work station entry for this work station. A sign-on display is shown, and any subsequent jobs from that work station continue to run in that subsystem, (unless another subsystem is started up that allocates the work station while it is at the sign-on display).
- If the user signs off and the subsystem in which his job was running is ended, the device is de-allocated. A second subsystem can then allocate the device and display the Sign On display.

Figure 3-3 on page 3-12 shows the subsystem activity required before the Sign-On display can appear.

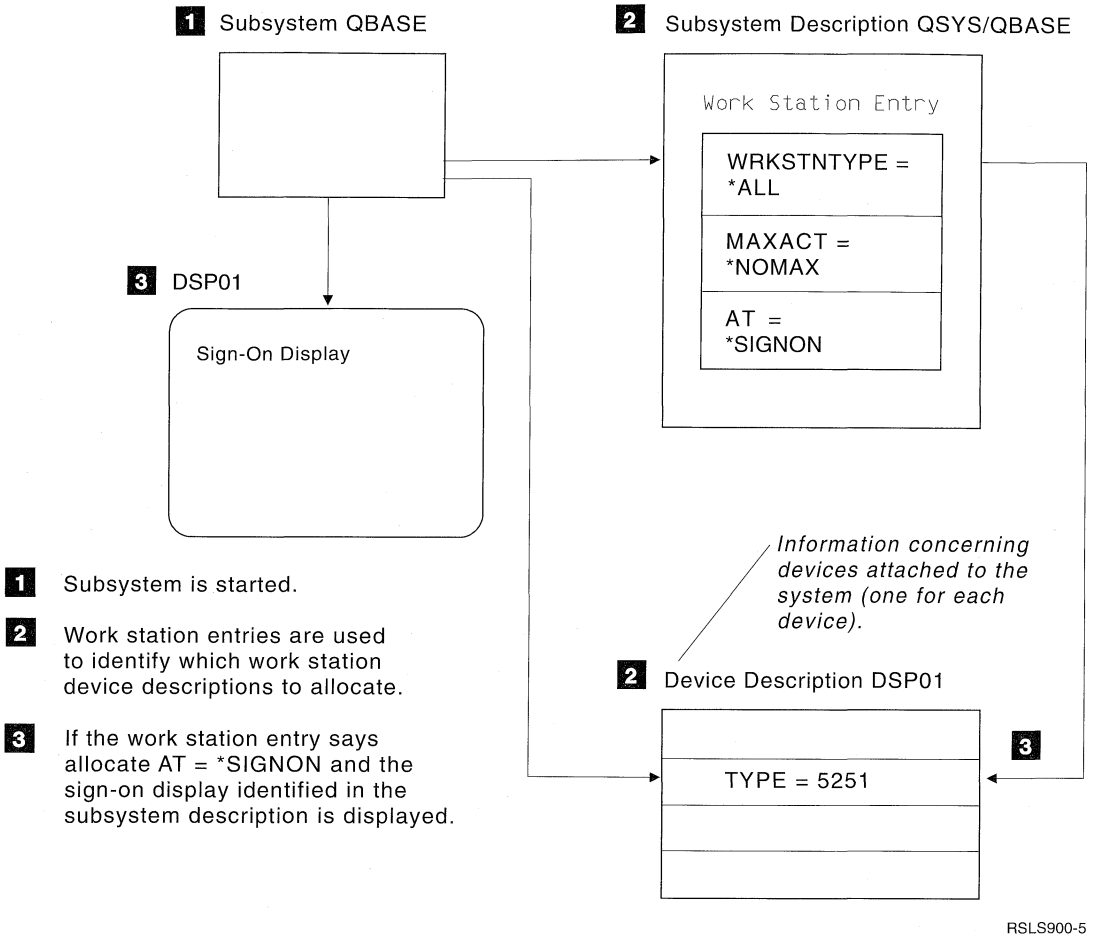


Figure 3-3. Subsystem Activity Leading to a Sign-On Display

The work station entries in the subsystem description QSYS/QBASE specify that jobs can be accepted from all work stations. When the QBASE subsystem is started, all online work stations are allocated to the subsystem (AT(*SIGNON) parameter) and the Sign On display is shown on each of the work stations.

Example of Allocating Work Station Devices

In this example, a subsystem A and subsystem B have work stations DSP01 and DSP02 in their subsystem description (the work station entries specify AT(*SIGNON)).

Device Name	Allocated to:
DSP01	Subsystem A
DSP02	Subsystem A

Assume that both work stations are varied on when subsystem A is started. Subsystem A allocates both work stations and shows the Sign On display on both. Even though subsystem A has the Sign On display shown on the work stations, they can be allocated by another subsystem or job; the work station would then no longer be available to subsystem A.

Device Name	Allocated to:
DSP01	USER1
DSP02	Subsystem A

When a user (USER1) signs on to work station DSP01, the device is allocated to USER1's job, which is running in subsystem A. Work station DSP02 is still at the Sign On display. Thus it can be allocated by another subsystem or job. It is then no longer available to subsystem A.

Device Name	Allocated to:
DSP01	USER1
DSP02	Subsystem B

Subsystem B is started. Because USER1 has signed on to work station DSP01, subsystem B cannot allocate the device. Subsystem B requests allocation of the device when it becomes available. DSP02 is allocated to subsystem B because no one had signed on to it in subsystem A. Any jobs started on DSP02 will run in subsystem B.

Device Name	Allocated to:
DSP01	Subsystem A
DSP02	Subsystem B

USER1 signs off. Because the user job was running in subsystem A, that subsystem displays the Sign On display so that another user can sign on the work station and run in subsystem A. If subsystem A is ended, work station DSP01 will be allocated by subsystem B (because it has an outstanding request to allocate the device.)

The name of the subsystem that currently has a work station allocated appears in the upper right corner of the Sign On display.

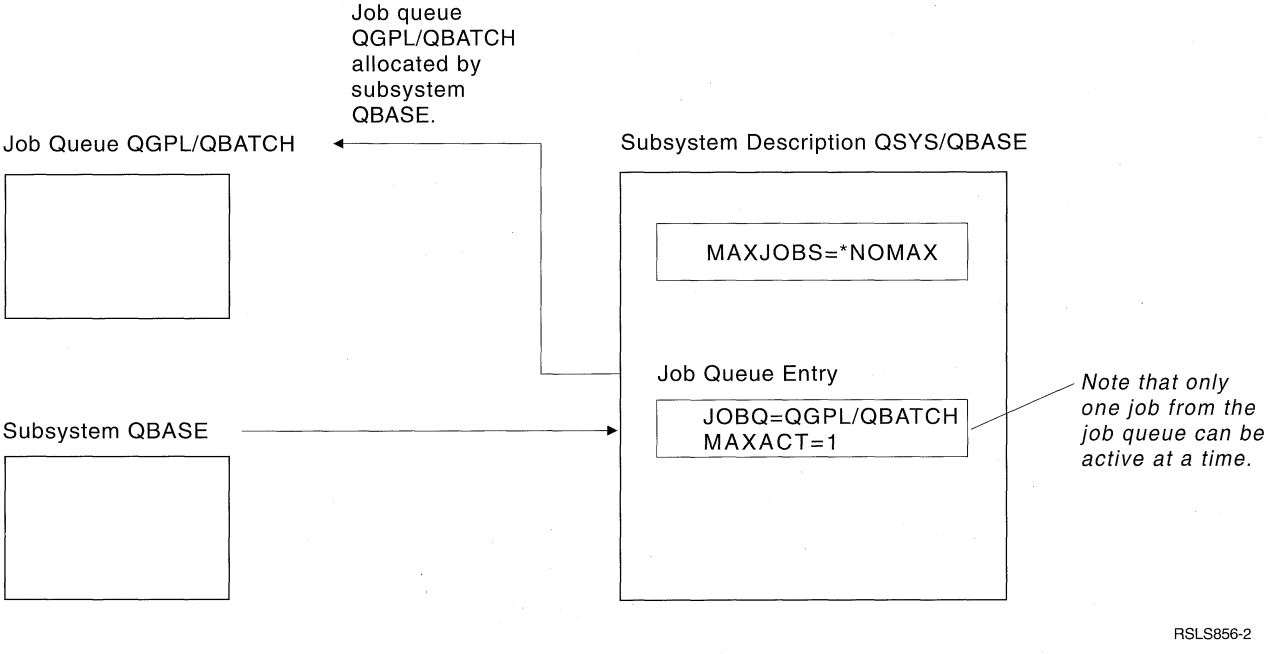


Figure 3-4. How the Job Queue Is Identified to the QBASE Subsystem

Allocating Job Queues

The primary difference between the interactive support on the AS/400 system and the support for batch jobs is that the batch functions are processed as a result of entries placed on a job queue instead of signing on to a work station.

Figure 3-4 shows how the subsystem QBASE, which is started using the IBM-supplied subsystem description QSYS/QBASE, operates for batch jobs.

A job queue entry in the subsystem description QSYS/QBASE specifies that jobs can be started using the job queue QGPL/QBATCH. Jobs can be placed on a job queue even if the subsystem has not been started. When the subsystem QBASE (or QBATCH, if using QCTL as the controlling subsystem) is started, it processes the jobs on the queue. A subsystem description can specify the maximum number of jobs (batch or interactive) that can be processed at the same time. For the job queue entry QBATCH in the subsystem QSYS/QBASE, only one batch job can be processed at any one time. (You can use the Change Job Queue Entry (CHGJOBQE) command

to change this at any time.) The number of jobs that can be active from any job queue is specified in the job queue entry.

Note that not all jobs on a job queue are necessarily available for processing when the subsystem is started; jobs can be held on a queue until the system operator releases them. If the subsystem is ended before all the jobs are processed, the jobs remain on the queue until the subsystem is started again or until another subsystem allocates the same job queue. More than one subsystem description can refer to the same job queue, but only one active subsystem at a time can use the job queue as a source of batch jobs. Therefore, if a subsystem ends and jobs are still on the job queue, another subsystem referring to that job queue can be started to process the jobs. If another subsystem is already started and is waiting for the same job queue, the subsystem automatically allocates the job queue when it becomes available.

Jobs can be placed on the job queue QGPL/QBATCH (or any job queue) in more than one way:

- A SBMJOB command can be used to submit a job to a user-specified job queue. The default of the command is to submit the job to run with the current user's user profile and some of the same job attributes. All other defaults are taken from a named job description or the currently active job (the job issuing the SBMJOB command). If no job description is specified, the job description specified in your user profile is used.
- Active jobs can be placed on the queue by specifying the TFRJOB or the TFRBCHJOB command in the active job.
- A SBMDBJOB command or SBMDKTJOB command can be used. For each job in the input stream, an entry is placed on a job queue. For example, the SBMDKTJOB command submits jobs from diskette to the job queue QGPL/QBATCH, which is the default. Jobs are delimited by BCHJOB and ENDBCHJOB commands.
- The system operator can issue a Start Database Reader (STRDBRDR) or Start Diskette Reader (STRDKTRDR) command to read input from a device or database file. For each job in the input stream, an entry is placed on a job queue. For example, the STRDBRDR command starts a reader to read records from a database member and places the job on the job queue QGPL/QBATCH. Jobs are delimited by BCHJOB and ENDBCHJOB commands.
- The Add Job Schedule Entry (ADDJOBSCDE) command can be used to place jobs on the job queue from a job schedule entry.

A batch job on any job queue can be moved to a different queue. For example, to move a job to the QBATCH job queue, type:

```
CHGJOB  JOB(xxxxx)  JOBQ(QBATCH)
```

When a job queue name is not specified on a Start Reader (STRxxxRDR) command or a Submit Jobs (SBMxxxJOB) command, the job is placed on the job queue QGPL/QBATCH. When a job queue name is not specified on a SBMJOB command, the job is placed on the job queue named in the job description specified by the SBMJOB command. Jobs submitted with the SBMDBJOB, SBMDKTJOB, or SBMJOB command can be displayed by using the Work with Submitted Job (WRKSBMJOB) command, unless DSPSBMJOB(*NO) was specified when the job was submitted.

When you submit a batch job, you can specify attributes of the job in one of two ways:

- Use a specified job description without overriding any of the attributes.
- Use a specified job description but override some of the attributes (using the BCHJOB or SBMJOB command) for the job.

The job description QGPL/QBATCH is the default for the BCHJOB command. The default user profile for the BCHJOB command is QPGMR because it is specified in the job description QGPL/QBATCH. The default user profile for the SBMJOB command is *CURRENT; the submitted job uses the same profile as the job that submitted it.

Allocating Communications Devices and Modes

When subsystems start, they request allocation of all communications devices in the communications entries in the subsystem description. The requests are sent to the QLUS (LU services) system job which handles device allocation for all communications devices. See Chapter 12, "System Jobs," for more information about QLUS.

QLUS gets notified when a communications device is available for program start request processing. This notification occurs when the connection between the local and remote system is established for that device. When QLUS receives this notification, it attempts to allocate the communications device to a subsystem based on communications entry definitions. If there is no subsystem active that wants to use the device, QLUS maintains allocation of the device until the device is varied off, or a subsystem starts that wants to use the device.

Rules for Device/Mode Allocation:

When more than one subsystem contains a communications entry for a communications device, QLUS uses the following rules to determine which subsystem will use the device (when the device is available):

- Communications entries with the highest level of detail for the device will be processed first. The order (from highest to lowest) of detail is: device name entry, remote location name entry, device type entry.
- Mode names are used only for advanced program-to-program communications (APPC)/advanced peer-to-peer networking (APPN) devices. Each mode on each device is allocated to a subsystem. A specific mode name will take priority over the generic *ANY mode name.
- When two or more subsystems have the same level of detail for the device and mode, the subsystem that first requested the device (the subsystem that was first started) will get to use the device.

Once a communications device is allocated to a subsystem it remains in that subsystem until the subsystem deallocates the device. When a subsystem deallocates a device (and deallocation is not due to a device error or varying off the device), QLUS attempts to allocate the device to another subsystem.

If a subsystem has a communications device allocated and you start a second subsystem that should really have the device (based on the device allocation rules), you can force the original subsystem to deallocate the device. This deallocation causes QLUS to go through the device allocation algorithm again, which will eventually cause the device to be allocated by the second subsystem. Here are some ways to cause a subsystem to deallocate a communications device:

- Varying the device off and then on again causes QLUS to attempt to allocate the device to a subsystem.
- Issuing the Allocate Object (ALCOBJ) command against the device works for some communications types (request a *EXCLRD lock). Issuing the Deallocate Object (DLCOBJ) command causes QLUS to attempt to allocate the device to a subsystem.
- Ending the subsystem (ENDSBS command) causes QLUS to automatically attempt to allocate the device to a subsystem.

The Controlling Subsystem

The controlling subsystem is identified in the system value QCTLSBSD. It is the interactive subsystem that starts automatically when the system starts, and is the subsystem through which the system operator controls the system. IBM supplies two complete controlling subsystem descriptions: QBASE (the default controlling subsystem) and QCTL. Only one controlling subsystem can be active on the system an any time.

The Restricted Condition: If the controlling subsystem is being ended, it goes into a restricted condition. The controlling subsystem is ended by either specifying its name or *ALL on the ENDSBS command or typing the ENDSYS command. To end the controlling subsystem, the ENDSBS or ENDSYS command is sent from an interactive job in the controlling subsystem, and only from a work station whose entry in the controlling subsystem description specifies AT(*SIGNON). The interactive job from which the command was issued remains active when the controlling subsystem goes into a restricted condi-

tion. If the job issuing the command is one of two jobs that are active at the work station (using the System Request key or the TFRSECJOB command), neither of the jobs is forced to end. However, the controlling subsystem does not end for the restricted condition until you end one of the jobs. Suspending group jobs also prevents the controlling subsystem from ending (until the group jobs are ended).

When the controlling subsystem is in the restricted condition, most of the activity on the system has ended, and only one work station is active. The system must be in this condition for commands such as Save System (SAVSYS) or Reclaim Storage (RCLSTG) to run. Some programs for diagnosing equipment problems also require the controlling subsystem to be in a restricted condition. To end this condition, you must start the controlling subsystem again.

How To's

This section tells you how to perform some common tasks involving subsystems.

Note: The IBM-supplied subsystem descriptions have been provided as examples and as backup for user-created subsystem descriptions. Therefore, we do not recommend modifying the subsystem descriptions in libraries QSYS and QGPL. You should make copies of the subsystem descriptions from these libraries and make changes to the copies.

Creating a Subsystem Description

You can create a subsystem description in two ways. You can copy an existing subsystem description and change it or you can create an entirely new description. The following are two approaches you can use:

1. Copying an existing subsystem description
   a. Create a duplicate object, CRTDUPOBJ, of an existing subsystem description. (You can also use the WRKOBJ or WRKOBJPDM commands.)
   b. Change the copy of the subsystem description.

   See Appendix B, "IBM-Supplied Object Contents," for examples.
2. Creating an entirely new subsystem description
   a. Create a subsystem description (CRTSBSD).
   b. Create a job description (CRTJOBD).
   c. Add work entries to the subsystem description.
      1) ADDWSE (Add work station entry)
      2) ADDJOBQE (Add job queue entry)
      3) ADDCMNE (Add communications entry)
      4) ADDAJE (Add autostart job entry)
      5) ADDPJE (Add prestart job entry)
   d. Create a class (CRTCLS).
   e. Add routing entries to the subsystem description (ADDRTGE).

Changing the Sign-On Display File

If you want to change the format of the Sign-on display, you can do the following:

1. Create a changed sign-on display file.
2. Change a subsystem description to use the changed display file instead of the system default of QSYS/QDSIGNON.
3. Test the change.
4. Change other subsystem descriptions.

Display File Source: The source for the sign-on display file is shipped as a member (QDSIGNON) in the QGPL/QDDSSRC physical file.

How to Change the Display File: There is a hidden field in the display file named UBUFFER. It is 128 bytes long and is stated as the last field in the display file. This field can be changed to function as an input/output buffer so the data specified in this field of the display will be available to application programs when the interactive job is started.

You can change the UBUFFER field to contain as many smaller fields as you need if the following requirements are met:

1. The new fields must follow all other fields in the display file. The location of the fields on the display does not matter as long as the order in which they are put in the data description specifications (DDS) meets this requirement.
2. The length must total 128. If the length of the fields is more than 128, some of the data will not be passed.
3. All fields must be input/output fields (type B in DDS source) or hidden fields (type H in DDS source).

Notes:

1. The order in which the fields in the sign-on display file are declared must not be changed. The position in which they are displayed on the display can be changed.
2. Do not change the total size of the input or output buffers. Serious problems can occur if the order or size of the buffers are changed.
3. Do not use the data descriptions specifications (DDS) help function in the sign-on display file.

Sign-On Display File Considerations:

The following suggestions apply to display files for sign-on displays:

- When creating the sign-on display file with the Create Display File (CRTDSPF) command, you must always specify 256 on the MAXDEV parameter.
- Help text must not be defined for sign-on displays.

Changing the Subsystem Description:

After the new sign-on display file has been created by using the CRTDSPF command, you can change the subsystem descriptions for subsystems that you want to use the new display by using the CHGSBSD command and specify the new display file on the SGNDSPF parameter. You should use a test version of a subsystem to verify that the display is valid before attempting to change the controlling subsystem.

Retrieving the Sign-On Information in an Application Program:

The data specified on the user portion of the Sign On display will be sent as replacement data in the CPF1124 message for interactive jobs. This data starts in byte 133 of the replacement text. The following program will retrieve the data:

```
PGM
DCL VAR(&MSGDTA)  TYPE(*CHAR) LEN(260) /*RECEIVES
    THE MESSAGE DATA */
DCL VAR(&MSGID)   TYPE(*CHAR) LEN(7)   /*RECEIVES
    THE MESSAGE ID   */
DCL VAR(&USERDTA) TYPE(*CHAR) LEN(128) /*RECEIVES
    THE USER DATA    */
/* Receive the CPF1124 message to obtain the user
    portion of the   */
/* Sign-On display. Note that the message is not
    removed so that  */
/* it will be available to the job log.  The
    message ID is also     */
/* retrieved to ensure that the message is CPF1124.
                */
RCVMSG    PGMQ(*EXT) RMV(*NO) MSGDTA(&MSGDTA)
   MSGID(&MSGID)
IF        COND(&MSGID *EQ 'CPF1124') THEN(CHGVAR
   VAR(&USERDTA) +
   VALUE(%SST(&MSGDTA 133 128)))
ENDPGM
```

Using the Retrieved Data:

After the retrieve program has run, the data that the user specified will be located in bytes 133 through 260 in the variable &MSGDTA. The CHGVAR command in the sample program moves the data to &USERDTA. The portions of the 128-byte buffer that were not used will appear as blanks in the variable &USERDTA.

Starting a Subsystem

To start a subsystem you can use the Start Subsystem (STRSBS) command by specifying the following:

```
STRSBS SBSD    (SBSD=library/subsystem
                  description name)
```

For example:

```
STRSBS MYLIB/MYSTORE
```

You can also start a subsystem using the Work with Subsystem Description (WRKSBSD) command.

Ending a Subsystem

To end a subsystem using the End Subsystem (ENDSBS) command, you must specify, using an option, when you want the subsystem to end. Use *IMMED to end the subsystem immediately, *CNTRLD to allow active jobs to end themselves (if they are checking to see if the job is being ended). To end a subsystem using the ENDSBS command, specify the following:

```
ENDSBS SBS OPTION  (SBS=the active subsystem
                   name)
```

For example:

```
ENDSBS MYSTORE *IMMED
```

Changing the Number of Jobs Allowed in a Subsystem

You can change the maximum number of jobs allowed in a subsystem by using the commands and their parameters:

Command	Parameter
CHGSBSD	MAXJOBS
CHGJOBQE	MAXACT
CHGWSE	MAXACT
CHGCMNE	MAXACT
CHGRTGE	MAXACT
CHGPJE	MAXJOBS

Changing the Size of a Storage Pool

One of the tasks you can perform when you do performance tuning is to change the size and activity level of the storage pools. See "Adjustments to Pool Sizes and Activity Levels" on page 13-13 for information on why and when you should consider adjusting pool sizes.

Machine and Base Pools: You can change the size of the machine pool using the CHGSYSVAL command or the WRKSYSSTS command. You can use the following for machine pools:

```
CHGSYSVAL QMCHPOOL 'new-size-in-KB'
```

This corresponds to pool 1 on the WRKSYSSTS display.

You can change the minimum size of the base pool using the CHGSYSVAL command. You can use the following for base pools:

```
CHGSYSVAL QBASPOOL 'new-minimum-size-in-KB'
```

This corresponds to pool 2 on the Work with System Status (WRKSYSSTS) display.

Note: The QBASPOOL system value only controls the minimum size of the base pool. The base pool contains all storage not allocated to other pools.

Other Storage Pools To change the size of a pool, you must first determine what the new size will be. Then use the Work with System Status (WRKSYSSTS) command to change the size of the pool.

Adding, Changing, or Removing Autostart Job Entries

The following commands allow you to add, change, or remove autostart job entries from a subsystem description. The subsystem may be active or inactive for these commands. For changes to take effect, the subsystem must be ended and started again.

- To add an autostart job entry to a subsystem description, use the Add Autostart Job Entry (ADDAJE) command. The following is an example of adding an autostart job entry:

  ```
  ADDAJE SBSD(USERLIB/ABC) JOB(START)
         JOBD(USERLIB/STARTJD)
  ```

- To specify a different job description for a previously defined autostart job entry, use the Change Autostart Job Entry (CHGAJE) command. The following is an example of changing an autostart job entry:

  ```
  CHGAJE SBSD(USERLIB/ABC) JOB(START)
         JOBD(USERLIB/NEWJD)
  ```

- To remove an autostart job entry from a subsystem description, use the Remove Autostart Job Entry (RMVAJE) command. The following is an example of removing an autostart job entry:

  ```
  RMVAJE SBSD(USERLIB/ABC) JOB(START)
  ```

Adding, Changing, or Removing Work Station Entries

The following commands allow you to add, change, or remove work station entries from a subsystem description. To use the following commands, the subsystem must be inactive.

- To add a work station entry to a subsystem description, use the Add Work Station Entry (ADDWSE) command. The following is an example of adding a work station entry:

```
ADDWSE SBSD(USERLIB/ABC) WRKSTN(DSP12)
       JOBD(USERLIB/WSE)
```

- To specify a different job description for a previously defined work station entry, use the Change Work Station Entry (CHGWSE) command. The following is an example of changing a work station entry:

```
CHGWSE SBSD(USERLIB/ABC) WRKSTN(DSP12)
       JOBD(USERLIB/NEWJD)
```

- To remove a work station entry from a subsystem description, use the Remove Work Station Entry (RMVWSE) command. The following is an example of removing a work station entry:

```
RMVWSE SBSD(USERLIB/ABC) WRKSTN(DSP12)
```

Adding, Changing, or Removing Job Queue Entries

The following commands allow you to add, change, or remove job queue entries from a subsystem description. To use the following commands, with the exception of the Change Job Queue Entry (CHGJOBQE) command, the subsystem must be inactive. The subsystem may be active when using the CHGJOBQE command.

- To add a job queue entry to a subsystem description, use the Add Job Queue Entry (ADDJOBQE) command. The following is an example of adding a job queue entry:

```
ADDJOBQE SBSD(USERLIB/ABC)
         JOBQ(USERLIB/NAME)
         MAXACT(*NOMAX)
```

- To specify a different job queue entry for a previously defined job queue entry, use the Change Job Queue Entry (CHGJOBQE) command. The following is an example of changing a job queue entry:

```
CHGJOBQE SBSD(USERLIB/ABC)
         JOBQ(USERLIB/NAME)
         MAXACT(12)
```

- To remove a job queue entry from a subsystem description, use the Remove Job Queue Entry (RMVJOBQE) command. The following is an example of removing a work station entry:

```
RMVJOBQE SBSD(USERLIB/ABC)
         JOBQ(USERLIB/NAME)
```

Adding, Changing, or Removing Communications Entries

The following commands allow you to add, change, or remove communications entries from a subsystem description. To use the following commands, the subsystem must be inactive.

- To add a communications entry in a subsystem description, use the Add Communications Entry (ADDCMNE) command. The following is an example of adding a communications entry:

```
ADDCMNE SBSD(COMMLIB/COMMSBS) DEV(*APPC)
        MAXACT(*NOMAX)
```

  Note: You must specify either the DEV parameter or the RMTLOCNAME parameter, but not both.

- To change a communications entry in a subsystem description, use the Change Communications Entry (CHGCMNE) command. The following is an example of changing a communications entry:

```
CHGCMNE SBSD(COMMLIB/COMMSBS) DEV(*APPC)
        MAXACT(12)
```

- To remove a communications entry from a subsystem description, use the Remove Communications Entry (RMVCMNE) command. The following is an example of removing a communications entry:

```
RMVCMNE SBSD(COMMLIB/COMMSBS) RMTLOCNAME(NYC)
```

  The following example removes only the entry with DEV(*ALL). It does not remove all the communications entries in the subsystem. For example,

```
RMVCMNE SBSD(COMMLIB/COMMSBS) DEV(*ALL)
```

Adding, Changing, or Removing Prestart Job Entries

The following commands allow you to add, change, or remove prestart job entries from a subsystem description. To use the following commands, with the exception of the Change Prestart Job (CHGPJE) command, the subsystem must be inactive. The subsystem may be active when using the CHGPJE command.

- To add a prestart job entry to a subsystem description, use the Add Prestart Job Entry (ADDPJE) command. The following is an example of adding a prestart job entry:

```
ADDPJE SBSD(USERLIB/ABC) PGM(START)
       JOBD(USERLIB/STARTPJ)
```

- To change a prestart job entry in a subsystem description, use the Change Prestart Job Entry (CHGPJE) command. The following is an example of changing a prestart job entry:

```
CHGPJE SBSD(USERLIB/ABC) PGM(START)
       JOBD(USERLIB/NEWPJ)
```

- To remove a prestart job entry from a subsystem description, use the Remove Prestart Job Entry (RMVPJE) command. The following is an example of changing a prestart job entry:

```
RMVPJE SBSD(USERLIB/ABC) PGM(START)
```

Adding, Changing, or Removing Routing Entries

The following commands allow you to add, change, or remove routing entries from a subsystem description. To use the following commands, the subsystem must be inactive.

- To add a routing entry to the subsystem description, use the Add Routing Entry (ADDRTGE) command. The following is an example of adding a routing entry:

```
ADDRTGE SBSD(USERLIB/ABC) SEQNBR(1)
        PGM(NAME) CMPVAL(*ANY)
```

- To specify a different routing entry for a previously defined routing entry, use the Change Routing Entry (CHGRTGE) command. The following is an example of changing a routing entry:

```
CHGRTGE SBSD(USERLIB/ABC) SEQNBR(1)
        CMPVAL(ACCOUNTING)
```

- To remove a routing entry from the subsystem description, use the Remove Routing Entry (RMVRTGE) command. The following is an example of removing a routing entry:

```
RMVRTGE SBSD(USERLIB/ABC) SEQNBR(1)
```

The IBM-Supplied QBASE and QCMN Subsystem Descriptions

Before program start requests are accepted by the system, a subsystem that supports communications must be started. There are two IBM-supplied subsystems, QBASE and QCMN, that accept program start requests for all communications types. QBASE is the default controlling subsystem and QCMN is the communications subsystem. Having either of these subsystems active allows program start requests to be accepted for all communications types.

The QCMN and QBASE subsystems have device type entries of *ALL and *ANY. Both subsystems have the appropriate routing entries (with CMPVAL(PGMEVOKE 29)) so all program start requests received by the system are accepted. If you have either of these subsystems and then start your own communications subsystem, or other subsystems such as DSNX(QDSNX) or SNADS(QSNADS), read the following section, "Allocating Communications Devices and Modes" on page 3-15. Both subsystems will be attempting to allocate the same communications devices.

Deleting a Subsystem Description

You can use the Delete Subsystem Description (DLTSBSD) command to delete a subsystem description. To use the DLTSBSD command, the subsystem cannot be active.

Adding a Second Job Queue

This example assumes that the system has already been changed to use the QBATCH subsystem for batch work.

The QBATCH subsystem is shipped with a single job queue for submitting batch jobs: QBATCH. A second job queue can be created and added to the subsystem.

To create the job queue, enter the following command:

```
CRTJOBQ   JOBQ(QGPL/QBATCH2)
          TEXT('Second job queue for QBATCH')
```

To add the job queue to the QBATCH subsystem when the subsystem is inactive, enter the following command:

```
ADDJOBQE  SBSD(QGPL/QBATCH)
          JOBQ(QGPL/QBATCH2)
          SEQNBR(20)
```

Allowing More Batch Jobs to Run

The QBASE subsystem is shipped with a job queue entry for the QBATCH job queue that only allows one batch job at a time to run. If you want to allow more batch jobs from that job queue to run at the same time, use the Change Job Queue Entry (CHGJOBQE) command.

The following command allows two batch jobs from the QBATCH job queue to run at the same time in the QBASE subsystem. You can issue this command at any time and takes effect immediately.

```
CHGJOBQE SBSD(QSYS/QBASE) JOBQ(QGPL/QBATCH)
         MAXACT(2)
```

Changing the Controlling Subsystem

Because the controlling subsystem is always active when the system is running, you cannot change its subsystem description. The following example assumes you are using QSYS/QBASE as the controlling subsystem description.

1. Create a copy of the subsystem description of the current controlling subsystem by using the Create Duplicate Object (CRTDUPOBJ) command.
2. Make the necessary changes to the copy.
3. Use the Work with System Value (WRKSYSVAL) command to change the controlling subsystem description (QCTLSBSD) to use the new subsystem description.
4. IPL the system.

Creating Another Controlling Subsystem

You may want to create another subsystem description for the controlling subsystem with attributes different from those in the shipped controlling subsystem description QSYS/QBASE. IBM also ships a QSYS/QCTL controlling subsystem. If you want to use it instead of QBASE, you just change the QCTLSBSD system value. See "Subsystem Configurations Shipped by IBM" on page C-2 for more information about the two subsystem configurations supplied by IBM.

IBM ships the QSYSSBSD subsystem in QSYS, which you cannot change. The subsystem can be used if your controlling subsystem becomes damaged or not usable.

If you create your own controlling subsystem description, you must change the system value QCTLSBSD. (QCTLSBSD contains the qualified name of the subsystem description for the controlling subsystem.) A change to this value takes effect at the next IPL unless the change is made by selecting "Define or change system at IPL menu" on the IPL options display during the IPL. In that case, the change takes effect during the current IPL.

You can use the subsystem description for the IBM-supplied controlling subsystem QBASE or QCTL as a model for creating your own controlling subsystem. For more information on the subsystem description for QBASE and QCTL, see Appendix B, "IBM-Supplied Object Contents."

The subsystem description for the controlling subsystem should contain a routing entry containing either *ANY or QCMDI as routing data, QSYS/QCMD as the program to be called, and class QSYS/QCTL or a user-defined class. This is because a user, usually the system operator, must be able to enter commands to do such things as free up storage if the auxiliary storage threshold has been reached.

The subsystem description for the controlling subsystem must contain a work station entry for the console device, and that entry must be of type *SIGNON. (*SIGNON is a value for the AT parameter, specified on the Add Work Station Entry (ADDWSE) command.) The *SIGNON value indicates that the sign-on display is dis-

played at the work station when the subsystem is started. This requirement ensures that the subsystem has an interactive device for entry of system and subsystem level commands. The End System (ENDSYS) command ends the OS/400 licensed program to a single session (or sign-on display) at the console in the controlling subsystem. A subsystem description that does not contain a work station entry for the console cannot be started as a controlling subsystem.

To have an alternative source of controlling input, the subsystem description for the controlling subsystem should have an entry for another work station. If a console problem is detected during an attended IPL and the system value QSCPFCONS is set to '1', the IPL will continue in unattended mode. Then, if the subsystem description for the controlling subsystem contains a work station entry for another work station, that work station can be used.

The subsystem description for the controlling subsystem should have a routing entry containing 525XTEST as routing data, QSYS/QARDRIVE as the program to be called, and QSYS/QCTL as the class. All work stations that are not allocated by subsystems or jobs have test request processed by the controlling subsystem.

After you have created the controlling subsystem, change the system value QCTLSBSD as follows (assuming the subsystem description is named QGPL/QCTLA):

```
CHGSYSVAL SYSVAL(QCTLSBSD) VALUE('QCTLA QGPL')
```

The change becomes effective at the next IPL.

If you create your own controlling subsystem, you should use a name other than QBASE or QCTL.

Controlling Security for Inactive Work Stations

You can request the subsystem to send a message to a message queue when an interactive job has been inactive for a specified period of time. A program monitoring that message queue, can then end the job if desired. You can also specify to have the system end or disconnect the job automatically. This control over inactive jobs provides security so that users do not leave signed-on terminals inactive.

The ADLINTJOBS parameter on the ENDJOB command allows you to end all of the interactive jobs associated with the work station, or all jobs associated with the group (if the job is a group job).

You can control the amount of time the work station can remain inactive before the subsystem sends a message (called time-out) by specifying a time interval in the QINACTITV system value. At the end of the specified time interval, the subsystem checks the interactive jobs running in it to determine whether any of the work stations have been inactive for the entire interval. An interactive job may complete up to double the time interval before a time-out occurs (the job is inactive for the entire current time interval plus the amount of time the job was inactive for the previous time interval. See "QINACTITV" on page 2-37 for a complete description of the system value QINACTITV.

A subsystem determines that a work station is inactive if all of the following are true:

- The job has not processed any additional transactions during the time interval.
- The job status is display wait.
- The job is not disconnected.
- The job status has not changed.
- The subsystem in which the job is running is not in the restricted state.

A transaction is defined as any operator interaction, like scrolling, pressing enter, pressing function keys, and so on. Typing at the work station without pressing enter is not considered a transaction. If a job at the work station, either secondary and/or group job, does not meet the inactive criteria, the job is considered active.

When an inactive job is found, the system value QINACTMSGQ is used to determine the processing options. The user can choose from the following processing options:

- Set the system value QINACTMSGQ to a message queue name. If you specify a message queue name for QINACTMSGQ, a user or program can monitor the message queue and take whatever action is necessary, such as ending the job.

  If a work station with a secondary job pair is inactive, two messages (one for each of the

secondary job pairs) are sent to the message queue. The user or program can then use the ENDJOB command against one or both of the secondary jobs.

If an inactive job has one or more group jobs, a single message is sent to each of the message queues. (The message queue indicates whether or not the job is a group job.)

A message continues to be sent for each interval that the job is inactive.

- Set the system value QINACTMSGQ to *ENDJOB. If you specify *ENDJOB for QINACTMSGQ, the system ends all of the jobs at the work station (both group and secondary jobs). A message indicating all jobs at the work station have been ended is sent to QSYSOPR.
- Set the system value QINACTMSGQ to *DSCJOB. If you specify *DSCJOB for QINACTMSGQ, the system disconnects all jobs at the work station (both group and secondary jobs). A message indicating all jobs at the work station have disconnected is sent to QSYSOPR.

Note: Source pass-through jobs and 3270 device emulation jobs are excluded from the time-out since they always appear inactive. System/36 Environment MRT jobs are also excluded because they appear as batch jobs.

Controlling User Sign-Ons

When a work station user signs on, the following actions occur:

- The routing table is searched for a match for the routing data, which comes from the job description.
- If QSYS/QCMD is the program specified in the routing entry that matched the routing data, the user profile is checked for an initial program. If an initial program is specified, that program is called. If control returns from the initial program, a check is made for an initial menu. If one is specified in the user profile, it is displayed.
- If QSYS/QCMD is not the program specified in the routing entry, the program that is specified is called.

Therefore, you can control what is displayed when a work station user signs on:

- You can create an initial program or initial menu that is specified in a particular user profile and specify QSYS/QCMD in the routing entry. Whenever the user signs on with this user profile, the same information is displayed regardless of which work station he uses.
- You can specify an initial program or an initial menu on the Sign On display if your user profile allows this.
- You can change the work station entry to specify a job description that has routing data to route to a particular routing entry and call a specific program so the routing operates the same regardless of which user signs on at the work station.
- You can use request data in a job description to call a program. In this case, if the work station user has an initial program associated with his user profile, that initial program is called. However, if there is no initial program associated with the user profile, the request data is processed and the specified program is called.

Using an Initial Program in a User Profile: To specify an initial program in a user profile, you use the Create User Profile (CRTUSRPRF) command:

```
CRTUSRPRF USRPRF(WSUSERS) INLPGM(USRMNUPGM)
          INLMNU(*SIGNOFF)
```

QSYS/QCMD should also be specified in the routing entry. The initial menu and initial program are called only when QSYS/QCMD is specified in the routing entry. If you route to a different program, the system does not call them. (You can call them with your own program. Use the RTVUSRPRF command to determine what they are.) Whenever someone signs on as WSUSERS, the user profile is checked for an initial program, and the program USRMNUPGM is called unless overridden on the sign-on display. This program can display a menu that allows work station users to select the application they want to run. By specifying the *SIGNOFF value on the initial menu (INLMNU) parameter, the system signs off the user when the initial program ends.

If you do not end the initial program, the program continues to run until the subsystem is ended or the job is ended.

For information about ending an initial program, see "Interactive Job Considerations" on page 5-7.

Using an Initial Menu in a User Profile:

To specify an initial menu in a user profile, you use the Create User Profile (CRTUSRPRF) command:

```
CRTUSRPRF USRPRF(WSUSERS) INLMNU(MAIN)
          INLPGM(SETUP)
```

QSYS/QCMD should also be specified in the routing entry. The initial menu and initial program are called only when QSYS/QCMD is specified in the routing entry. If you route to a different program, the system does not call them. (You can call them with your own program. Use the RTVUSRPRF command to determine what they are.) Whenever someone signs on as WSUSERS, the user profile is checked for an initial program, and the program SETUP is called unless overridden on the sign-on display when the program setup is done. The initial menu MAIN is displayed (unless overridden on the sign-on display). The initial menu continues to be displayed until you sign off.

Using Routing Data:

When you use routing data to determine a routing program for a work station, you need to create a job description that specifies the routing data and a job description in the user profile, and add a routing entry to the subsystem description. (To change the subsystem description, the subsystem must be inactive.)

```
CRTJOBD JOBD(DSP02JOBD) USER(*RQD)
        RTGDTA('DSP02')
ADDWSE SBSD(QINTER) WRKSTN(DSP02)
       JOBD(DSP02JOBD)
ADDRTGE SBSD(QINTER) SEQNBR(100)
       CMPVAL('DSP02')
        PGM(DSP02MNU) POOLID(2)
```

Whenever anyone signs on at work station DSP02, the system retrieves the routing data from the associated job description (DSP02JOBD) and compares it with the compare values specified in the routing entries. Because the routing data matches the compare value in the routing entry that was added to the subsystem description for QINTER, the program DSP02MNU is called.

Using Request Data:

When you use request data to determine a controlling program for a work station, you need to create a job description that specifies the request data, add a work station entry to the subsystem description specifying this job description, and create a user profile that specifies the password to be used.

```
CRTJOBD JOBD(DSP03JOB) USER(*RQD)
        RQSDTA('CALL DSP03MENU')
        RTGDTA(QCMDI)
ADDWSE SBSD(QINTER) WRKSTN(DSP03)
       JOBD(DSP03JOBD)
CRTUSRPRF USRPRF(DSP03USER)
          INLPGM(QCMD)
```

If someone signs on at work station DSP03 with user name DSP03USER, the program DSP03MENU is called. If DSP03USER is used to sign on at another work station that does not have a special work station entry set up, the initial menu MAIN is displayed (because MAIN is the default for INLMNU on the CRTUSRPRF command).

Using Double-Byte Request Data:

You can use double-byte data as part of the request data entered with the following commands:

- Reroute Job (RRTJOB)
- Transfer Job (TFRJOB)
- Transfer Batch Job (TFRBCHJOB)

The number of double-byte characters that you can include in request data, including shift control characters, is half the allowed number of alphanumeric characters.

Note: Do not use double-byte data as request data (RQSDTA) or routing data (RTGDTA) parameters on the Batch Job (BCHJOB) and Submit Job (SBMJOB) commands because the system might not properly process the data.

Preventing Request End or Sign-Off:

Normally, a work station user can end a request or sign off by requesting the System Request menu and selecting the appropriate option. This could cause a problem if the work station user ended a request or signed off in the middle of a multiple operation transaction.

You can prevent access to the System Request menu (the *Security Reference* manual contains information on how to do this). You can also allow access to the System Request menu, but prevent the use of certain commands. For example, the security officer can revoke public authority to the End Request (ENDRQS) and SIGNOFF commands and give it only to those users that are not running sensitive applications. For the work station user to sign off, the user must call a CL program that issues the SIGNOFF command. This program must be owned by someone with authority to the command and must be created with the value USRPRF(*OWNER) specified on the Create CL Program (CRTCLPGM) command.

If the work station user's authority to the ENDRQS and SIGNOFF commands is revoked and the user attempts to select these options from the System Request menu, a message is received saying the user does not have authority to the commands.

Preventing Sign-On Display from QINTER

This example assumes that the system uses the QINTER subsystem for interactive work.

If you do not want the sign-on display shown at some work stations when you start the QINTER subsystem, you can do the following:

Add the work station entries in the QSYS/QINTER subsystem description to specify AT(*ENTER) in the work station name (WRKSTN) entry for work stations where you do not want the sign-on display to appear. This allows interactive jobs to be transferred to the QSYS/QINTER subsystem. You should not change the WRKSTNTYPE(*ALL) entry.

Controlling a Group of Work Stations

You may want to control a group of work stations separately. For example, if you have a group of work station users who are using the same application, you may want that group of users to sign off to perform some special application function or to send them a message. You can:

- Create a separate subsystem for the group of work station users. A separate subsystem allows a convenient start and shutdown, but it does not simplify the job of sending messages to the group.
- Create a command or a CL program that can be used to send a message to a group of work stations. However, this approach does not assist in controlling sign-on or sign-off. See the *CL Programmer's Guide*, for an example of a CL program that can be used to send a message to a group of work stations.

Controlling Initial Program Load (IPL)

This section describes ways in which you can control the IPL.

Changing the IPL Start-Up Program:

The autostart job in the controlling subsystem transfers control to the program specified in the system value QSTRUPPGM. You can tailor this program.

You can create your own program and change the QSTRUPPGM system value to that program name. Or, you can use the shipped program QSTRUP in QSYS as a base to create your own program. To do this, you would:

1. Retrieve the source of the shipped program using the RTVCLSRC command (for example, `RTVCLSRC PGM(QSYS/QSTRUP) SRCFILE(YOURLIB/YOURFILE)`).
2. Change the program.
3. Create the program using the CRTCLPGM command, putting it into your own library.
4. Test the program to ensure that it works.
5. Change the system value QSTRUPPGM to the program name and library you specified on the CRTCLPGM command.

Calling a Special IPL Recovery Program:

To call a special recovery program if the IPL senses that the previous system ending was abnormal, you can add an autostart job entry to the subsystem description for the controlling subsystem. This program checks the system value QABNORMSW. For a normal system ending, the value of QABNORMSW is '0', and for an abnormal system ending the value of

QABNORMSW is '1'. See the example program in Figure 3-5 on page 3-26.

```
SEQNBR *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7
  1.00 /* SPCRECOV - Autostart program to call special recovery program */
  2.00              PGM
  3.00              DCL         &QABNORMSW *CHAR LEN(1)
  4.00              RTVSYSVAL   SYSVAL(QABNORMSW) RTNVAR(&QABNORMSW)
  5.00              IF          (&QABNORMSW *EQ '1') DO /* Recover */
  6.00              SNDPGMMSG   MSG('Recovery program in operation-do not +
  7.00                            start subsystems until notified') +
  8.00                            TOMSGQ(QSYSOPR)
  9.00              CALL        RECOVERY
 10.00              SNDPGMMSG   MSG('Recovery complete-jobs may be started') +
 11.00                            TOMSGQ(QSYSOPR)
 12.00              ENDDO       /* Recover */
 13.00              ENDPGM
```

Figure 3-5. Special IPL Recovery Program

An alternative would be to drop the messages and start up other subsystems when your recovery function is complete.

Setting Up an Unattended Environment:

To set up an unattended environment and prevent unauthorized users from using the console while the system is unattended, have the system operator change the QSYSOPR message queue to default delivery and sign off. Signing-off prevents unauthorized users from using the console. The following command changes the QSYSOPR message queue:

```
CHGMSGQ MSGQ(QSYSOPR) DLVRY(*DFT)
```

Setting Up an Unattended Nighttime Environment:

You may have jobs in your installation that can be run in an unattended nighttime environment. These jobs could be submitted throughout the day to be processed at night. To set up this environment, you could create a special subsystem for the nighttime work. The subsystem description and job queue could be defined with the following commands:

```
CRTSBSD     SBSD(QGPL/NIGHTQ) POOLS((1 *BASE))
            TEXT('Nighttime jobs')
CRTJOBQ     JOBQ(QGPL/NIGHTQ) TEXT('Nighttime
            job queue')
ADDJOBQE    SBSD(QGPL/NIGHTQ) JOBQ(QGPL/NIGHTQ)
            MAXACT(1)
ADDRTGE     SBSD(QGPL/NIGHTQ) SEQNBR(10)
            CMPVAL(*ANY) PGM(QSYS/QCMD)
            CLS(QGPL/QBATCH)
```

You then create a job description using the Create Job Description (CRTJOBD) command that specifies the job queue QGPL/NIGHTQ on the JOBQ parameter.

When ready to start the NIGHTQ subsystem, the system operator:

1. Ends all active subsystems except the controlling subsystem using the End Subsystem (ENDSBS) command.
2. Starts the NIGHTQ subsystem with the Start Subsystem (STRSBS) command.
3. Changes the QSYS/QSYSOPR message queue to place it in default mode so any messages requiring a response receive a default reply.
4. Submits a job with request data of a Power Down System (PWRDWNSYS) command and a job priority of 9 to the NIGHTQ job queue. This priority places the job at the end of the job queue so the system will be powered down when all the jobs in the job queue have been processed.
5. Signs off.

The STRSBS command, the Change Message Queue (CHGMSGQ) command, and the PWRDWNSYS command could be placed in a CL program to simplify the operator's job. The CL program could be:

```
PGM
CHGMSGQ    QSYSOPR DLVRY(*DFT)
STRSBS     NIGHTQ
SBMJOB     JOBD(QBATCH) JOBQ(QGPL/NIGHTQ)
           JOBPTY(9)
           CMD(PWRDWNSYS *IMMED)
SNDPGMMSG MSG('NIGHTQ subsystem started and
           power down job submitted')
ENDPGM
```

This solution assumes that only one job is run at a time from the NIGHTQ job queue and that no other jobs are running while the nighttime subsystem is running. If this is not desirable, the procedure should be changed as in the following example so that two jobs run at once and when they are completed, the system powers down.

The following CL program could be used to start the NIGHTQ subsystem with two batch jobs running and to ensure that all jobs have finished before the system is powered down. The job queue entry shown in the previous example should be changed to meet your requirements.

```
PGM
CHGMSGQ QSYSOPR DLVRY(*DFT)
CHGSBSD QGPL/NIGHTQ MAXJOBS(2) /* Reset to 2 */
STRSBS NIGHTQ
SBMJOB JOBD(QBATCH) JOBQ(QGPL/NIGHTQ) JOBPTY(9)
       CMD(CHGSBSD QGPL/NIGHTQ MAXJOBS(1))
       JOB(CHGMAXJOBS)
SBMJOB JOBD(QBATCH) JOBQ(QGPL/NIGHTQ) JOBPTY(9)
       CMD(PWRDWNSYS *IMMED) JOB(POWERDOWN)
SNDPGMMSG MSG('NIGHTQ subsystem started and
       power down job submitted')
ENDPGM
```

When the subsystem is started, the MAXJOBS value is set to 2. The first job submitted (CHGMAXJOBS) then sets the MAXJOBS value to 1 just before the power down job is run. Because the priority of both the CHGMAXJOBS and POWERDOWN jobs is 9, neither job runs until all higher priority jobs in the queue (1 being the highest priority) are run.

Mixing Double-Byte and Alphanumeric Display Stations

If you use both double-byte and alphanumeric devices with your system, you can create an initial program to have double-byte versions of user-written alphanumeric messages and user-designed displays shown at double-byte work stations. The following procedure describes how to show the double-byte versions of messages and displays at double-byte work stations:

1. Create a library for storing the double-byte versions of messages and display files. For example, to create a library named DBCSLIB, enter the following command:

   ```
   CRTLIB LIB(DBCSLIB) TEXT('Double-byte
          version of objects')
   ```

2. Put a copy of the messages and display files into the library you just created.

3. Translate the information in the copied file from alphanumeric information to double-byte information. For example, change message text from alphanumeric to double-byte.

   Do not change the file name.

4. Create a job description to be used for the double-byte devices. This job description should be similar to that used for alphanumeric work stations. However, the job description has different routing data. For example, to create the job description DBCSJOBD, enter the following command:

   ```
   CRTJOBD JOBD(DBCSJOBD) RTGDTA(DBCSWS)
           LOG(4 0 *SECLVL) TEXT('Double-byte
           work station job description')
   ```

5. Create a copy of the subsystem description. This description is changed for use as a subsystem description for DBCS work stations. For example, to create a copy of the subsystem description named QINTER as the subsystem description DBCSSBSD, enter the following command:

   ```
   CRTDUPOBJ OBJ(QINTER) FROMLIB(QSYS)
             OBJTYPE(*SBSD) NEWOBJ(DBCSSBSD)
   ```

6. Change the subsystem description so that DBCSLIB is added ahead of the other libraries in the system library list. For example:

   ```
   CHGSBSD SBSD(DBCSLIB) SYSLIBLE(DBCSLIB)
   ```

7. Change the work station entry for the 5555 Display to specify the job description you just created. For example, to specify a job description named DBCSJOBD, enter the following command:

   ```
   CHGWSE SBSD(DBCSSBSD) WRKSTNTYPE(5555)
          JOBD(DBCSJOBD)
   ```

8. Add a routing entry that matches the routing data of the double-byte job description. For example:

```
ADDRTGE SBSD(DBCSSBSD) SEQNBR(15)
        CMPVAL(DBCSWS) PGM(DBCSPGM)
```

9. Create a CL program named DBCSPGM, similar to the one shown below that overrides the default printer file so that the file is double-byte. When the default printer file is double-byte, the system properly prints double-byte output printed as a result of pressing the Print key.

```
PGM
   OVRPRTF FILE(QSYS/QSYSPRT) IGCDTA(*YES)
   TFRCTL QSYS/QCMD
ENDPGM
```

When you use the subsystem description DBCSSBSD, the double-byte versions of user-written messages and user-designed displays that are stored in the library DBCSLIB are shown at double-byte work stations. The alphanumeric versions of these messages and displays continue to be shown at alphanumeric work stations in the QINTER subsystem.

Interactive Subsystem Example

In the following example, objects are created for a subsystem in which interactive work is to be processed. The subsystem automatically allocates two work stations DSP01 and DSP02 when it is started. When a work station user signs on, program ORDLIB/ORDERPGM is called. Class QGPL/ORDCLS contains the running parameters for the routing step, and the job attributes are contained in the job description specified in the user profile.

The subsystem description ORDER is created and placed in library QGPL. When the subsystem ORDER is started, one storage pool with a size of 500KB is allocated. The pool's activity level is 2. The subsystem monitor programs called to handle the jobs in the subsystem ORDER use storage from pool 1, which is the shared base storage pool, but use machine pool activity level. The following CRTSBSD command creates the subsystem description ORDER:

```
CRTSBSD   SBSD(QGPL/ORDER)
          POOLS((1 *BASE) (2 500 2))
          TEXT('Order entry jobs')
```

For an example of an autostart job in a controlling subsystem, see the example on "Calling a Special IPL Recovery Program" on page 3-25, or the QBASE and QCTL subsystem descriptions in Appendix B, "IBM-Supplied Object Contents" on page B-1.

The following command creates the class ORDCLS and places it in QGPL:

```
CRTCLS    CLS(QGPL/ORDCLS) RUNPTY(45)
          TEXT('Class for order entry')
```

The following commands add two work station entries to the subsystem description QGPL/ORDER. When the subsystem ORDER is started, DSP01 and DSP02 are allocated to ORDER.

```
ADDWSE  SBSD(QGPL/ORDER) WRKSTN(DSP01)
ADDWSE  SBSD(QGPL/ORDER) WRKSTN(DSP02)
```

The following command adds a routing entry to the subsystem description QGPL/ORDER. The interactive jobs started from DSP01 and DSP02 are automatically routed to a routing step in which program ORDLIB/ORDERPGM is first called.

```
ADDRTGE SBSD(QGPL/ORDER) SEQNBR(010)
        CMPVAL(QCMDI) PGM(ORDLIB/ORDERPGM)
        CLS(QGPL/ORDCLS) POOLID(2)
```

The following command should be specified for any subsystem that has work station entries. Without it, you cannot use the Test Request key on the 5250 work stations allocated to the subsystem. (The Test Request key is used to start verification tests for a work station.)

```
ADDRTGE   SBSD(QGPL/ORDER) SEQNBR(900)
          CMPVAL(525XTEST) PGM(QSYS/QARDRIVE)
          CLS(QSYS/QCTL)
```

Note: Because the POOLID parameter was not specified, a test request job runs in pool 1 of the subsystem ORDER. This pool is defined as the base storage pool.

Chapter 4. Jobs

This chapter includes information that is common to all jobs on the AS/400 system, like how jobs are started and routed, attributes of a job, and job logs. "Job Commands" on page 4-2 provides a quick reference for common commands that affect jobs on the system. Information unique to each job type is found in Chapter 5 through Chapter 12.

Basic Job Types

The two basic types of jobs are interactive jobs and batch jobs.

- Interactive jobs occur when a user signs on to a work station, transfers to a secondary or group job, or presses the Test Request key. These are jobs that require interaction with a user at a work station. Interactive jobs are further defined as the following (based on how they are started and how the system handles them):
  - Interactive jobs
  - Group jobs
  - Secondary jobs
- Batch jobs occur by a user submitting a job to a job queue, issuing a communications program start request, starting the subsystem with an autostart job entry, or starting the subsystem with a prestart job entry. These jobs allow work to occur without constant interaction with a user. Two examples of jobs that are commonly run in batch are compiling programs and running reports. Batch jobs are further defined as the following (based on how they are started and how the system handles them):
  - Batch jobs
  - Spooling jobs
  - Communications jobs
  - Autostart jobs
  - Prestart jobs
  - System jobs

Job Names

To make it easier to control and identify jobs on the system, each job has a unique qualified job name. The **qualified job name** consists of three parts: the job name, the user name, and the job number.

- For interactive jobs, the **job name** is the same as the name of the work station you signed on to. For batch jobs you can specify your own job name. The job name can be up to 10 characters long.
- The **user name** is the name of the user profile under which the job is started. For interactive jobs, the user name is the name you entered in the user field on the sign-on display. For batch jobs you can specify the user profile under which the batch job is to run. The user name can be up to 10 characters long.
- The **job number** is a unique number assigned by the system so you can identify jobs, even if more than one has the same job name and user name. The job number is always 6 numeric digits.

The syntax for qualified job names is similar to qualified names for objects. For example, if the job name is DSP01, the user is QPGMR, and the job number is 000578, the qualified job name is entered on the Work with Job (WRKJOB) command as follows:

```
WRKJOB JOB(000578/QPGMR/DSP01)
```

Another similarity to object names is that you do not need to specify all of the qualifiers. For example you could specify the following:

```
WRKJOB JOB(QPGMR/DSP01)
```

or

```
WRKJOB JOB(DSP01)
```

This works the same as entering the entire qualified job name. If several (more than one) jobs on the system match the portion of the job name you entered, the Select Job display appears. This display allows you to select which job you want from a list of duplicate job names.

Job Commands

Figure 4-1 on page 4-2 provides a quick reference to some of the most commonly used job commands.

Figure 4-1 (Page 1 of 2). Job Commands

Command	Description
BCHJOB	Indicates the beginning of a batch job in a batch input stream.
CHGJOB	Changes attributes including job queue, print device, running priority, and more.
DATA	Used in an input stream to indicate the beginning of an inline data file.
DLYJOB	Allows the current job to wait for a specified number of seconds, or time of day, before running resumes.
DMPJOB	Dumps the basic data structures or specific calls of the current job or of the job being serviced as a result of the Start Service Job (STRSRVJOB) command.
DSCJOB	Disconnects all interactive jobs at the work station and returns to the sign-on.
DSPJOB	Shows information about a job, including status, job definition, run attributes, job log, and more.
DSPJOBLOG	Shows commands and related messages for a job when its job log has not been written.
ENDBCHJOB	Acts as a delimiter in a batch input stream indicating the end of a job.
ENDGRPJOB	Ends a single job within a group and resumes another job within the group.
ENDJOB	Ends a job.
RRTJOB	Causes a new routing step to start for a job in the current subsystem.

Figure 4-1 (Page 1 of 2). Job Commands

Command	Description
RTVJOBA	Used in a CL program to retrieve the values of one or more job attributes and place the values into the specified CL variable.
SBMJOB	Allows a job that is running to submit another job to a job queue to run later in batch.
SIGNOFF	Ends an interactive job or causes all jobs in a group to end.
STRDBRDR	Starts a spooling reader using a database file.
STRDKTRDR	Starts a spooling reader to the specified diskette unit to read a batch input stream and place it on the appropriate job queue.
TFRBCHJOB	Transfers a batch job to the specified job queue.
TFRGRPJOB	Suspends the job that issued the TFRGRPJOB command and the group job specified by the Group job prompt (GRPJOB parameter) is resumed or is started.
TFRJOB	Transfers a job to the specified job queue that is run in the subsystem where that queue is active.
TFRSECJOB	Creates a secondary interactive job at your work station, then transfers control between the primary and secondary jobs.
WRKACTJOB	Works with performance and status information for active jobs in the system.
WRKJOB	Allows you to work with or change information concerning a user job, including job status, job definition attributes, run attributes, job log information, and more.
WRKSBMJOB	Allows you to work with all jobs submitted from your work station, job, or user profile.

Figure 4-1 (Page 2 of 2). Job Commands

Command	Description
WRKSBSJOB	Allows you to work with jobs running in subsystems or jobs that are on a job or output queue.
WRKUSRJOB	Provides a list of selected user jobs and supports options to work with each job.

Job Starting and Routing

When a job, other than a prestart job, is started, the subsystem that is starting the job must determine what the first program is that the job will run. This is called *routing the job*. As was shown under "Routing Entries and Routing Data" on page 3-9, this is done by attempting to match up the routing data for the job with a routing entry in the subsystem description. A prestart job runs the program that is specified on the prestart job entry. Each time that a job causes a subsystem to search the routing table (for any reason), it adds a **routing step**. A routing step is the processing that results from running a program specified in a routing entry. Most jobs will only have one routing step, but it is possible for some jobs to have many routing steps. The following list identifies all of the possible ways to cause the job to have another routing step:

- Reroute Job (RRTJOB). This causes the job to reroute in the subsystem that the job was started in.
- Pressing F3 from the highest level of the command entry display (only valid for interactive jobs).
- Return (RETURN) command from the highest program in the job's program stack (only for interactive jobs—if a batch job issues the Return command, the batch job will end).
- TFRJOB and TFRBCHJOB commands. This actually causes the job to be placed on a job queue and then rerouted in the subsystem that has the job queue allocated.

Note: Prestart jobs do not use routing data or routing entries when they are started. Prestart jobs also cannot be rerouted (RRTJOB) or transferred (TFRJOB).

To summarize, a job can have one or many routing steps, but all of the routing steps belong to that job. Normally, a job has only a single routing step.

Each routing step in a started job operates under the control of the program called at the start of the routing step. The routing data received by the system determines the routing entry in which the program name can be found.

Note: The IBM-supplied routing program QCMD runs the initial program and menu for interactive users. If you define your own routing programs, you should consider running a user's initial program and/or menu.

Routing data can be specified in one of the following ways:

- In the job description for the job (RTGDTA parameter in CRTJOBD command), or RQSDTA parameter, if *RQSDTA is specified for the RTGDTA parameter.
- In the RTGDTA parameter specified in a BCHJOB or SBMJOB command (overrides the routing data in the job description) or RQSDTA parameter, if *RQSDTA is specified for the RTGDTA parameter.
- In the RTGDTA parameter specified on a Reroute Job (RRTJOB), Transfer Job (TFRJOB), or Transfer Batch Job (TFRBCHJOB) command (instead of the routing data in the job description) or RQSDTA parameter, if *RQSDTA is specified for the RTGDTA parameter. The RRTJOB, TFRJOB, and TFRBCHJOB commands can be run in a job after the job was initially routed using one of the previously described ways of specifying routing data.
- Routing data is created by the subsystem as a result of a communications program start request being processed. Routing data is based on information supplied by the remote system in the program start request (for example, program name and library).

For an online list of all the commands that can be used with jobs, type `go cmdjob` on the command line.

Job Attributes (Job Description Object)

A job description is used to collect many of the parameters used when starting a job. The job description gets rid of the need to specify the same parameters repeatedly for each job. You create a job description by using the Create Job Description (CRTJOBD) command. The following example and table show how you can create one job description and use it so you do not have to specify several parameters in a lot of commands:

```
CRTJOBD  JOBD(BATCH4) USER(*RQD) JOBPTY(4)
         OUTPTY(4) RTGDTA(QCMDI)
```

Command	Parameter	Value from JOBD on CRTJOBD Command
ADDAJE	JOBD	BATCH4
ADDCMNE	JOBD	BATCH4
BCHJOB	JOBD	BATCH4
CRTUSRPRF	JOBD	BATCH4
SBMJOB	JOBD	BATCH4

The following list describes some of the parameters on the CRTJOBD command that can be specified that are important to work management:

- User name (USER). The default for the user profile the job is to use.
- Job queue (JOBQ). The job queue on which the job is to be placed (batch job only).
- Job priority (JOBPTY). Priority of the job on the job queue (batch job only).
- Output priority (OUTPTY). Priority of any spooled output files created by the job.
- Routing data (RTGDTA). Data used to determine which routing entry to use for the job.
- Request data (RQSDTA). Data to be put on the job message queue.
- Initial library list (INLLIBL). The user part of the initial library list for the job.
- Printer device (PRTDEV). The default printer device on which the spooled output files made by the job are to be printed.
- Output queue (OUTQ). The default output queue on which the spooled output files made by the job are to be placed. (Files are placed on this queue if OUTQ(*JOB) is specified for the file or in an override command.) This defaults to the output queue associated with the printer device.

When you define a job, you can use the job description in one of two ways:

- Use a specified job description without overriding any of its attributes. For example:

  ```
  SBMJOB JOB(OEDAILY) JOBD(QBATCH)
  ```

- Use a specified job description but override some of the attributes (using the BCHJOB or SBMJOB command). For example, to override the message logging in the job description QBATCH, you specify:

  ```
  SBMJOB JOB(OEDAILY) JOBD(QBATCH)
         LOG(2 20 *SECLVL)
  ```

You cannot override any job description attributes for autostart jobs, work station jobs, or communications jobs.

The following commands can also be used with job descriptions:

- Change Job Description (CHGJOBD). Changes the contents of a job description.
- Delete Job Description (DLTJOBD). Deletes a job description.
- Display Job Description (DSPJOBD). Displays the contents of a job description.
- Work with Job Descriptions (WRKJOBD). Allows you to change, copy, delete, or display a job description.

Figure 4-2 on page 4-5 shows how the job description and class relate to the subsystem description.

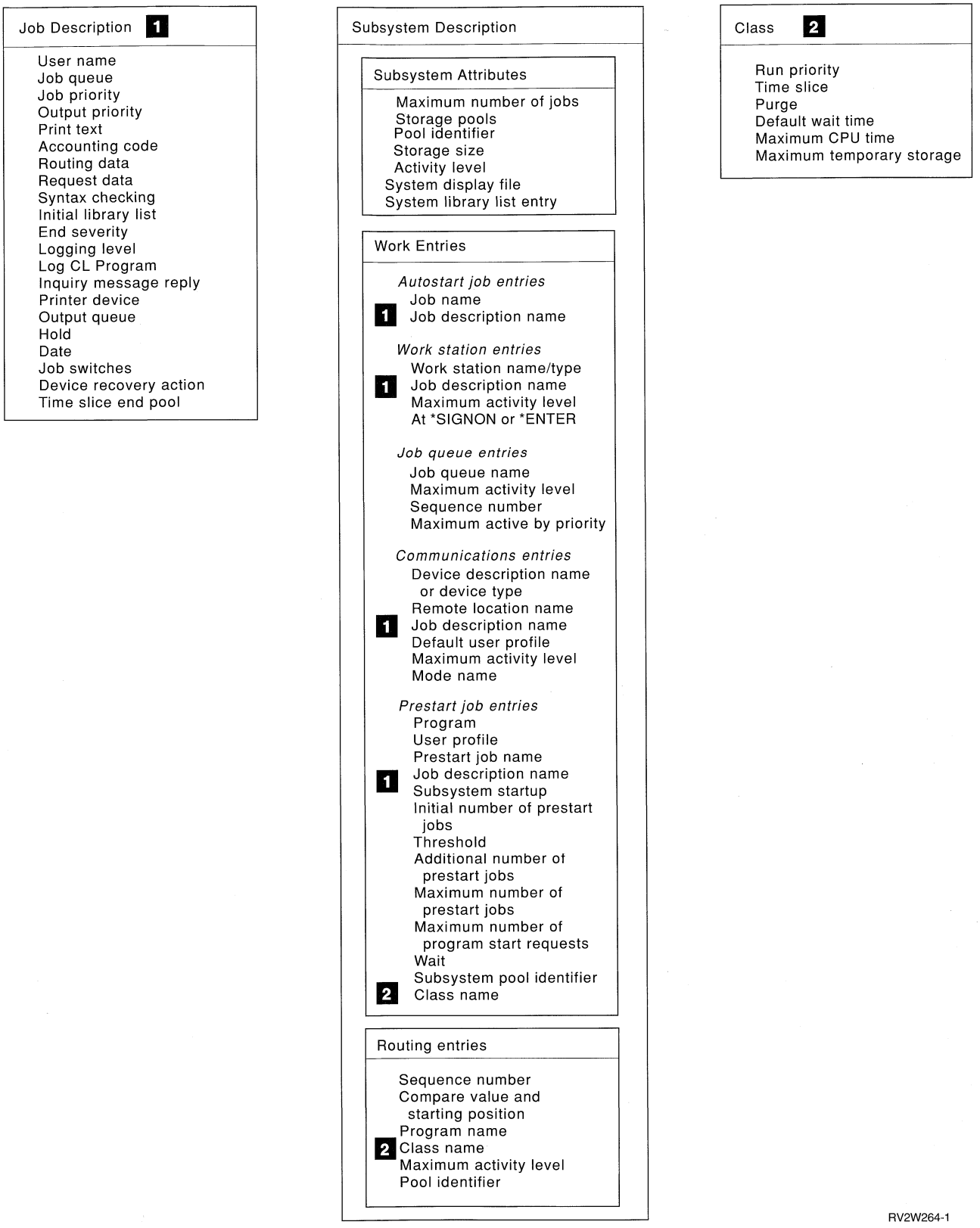


Figure 4-2. Relationships among Job, Class, and Subsystem Descriptions

Security Considerations for Job Descriptions

A job description that has a user profile name (USER) specified should be authorized only to specific individuals. If not, other users will have access to that job description when running their jobs. For example, if you specified:

```
CRTJOBD JOBD(XX) USER(JONES) . . . AUT(*USE)
```

In this example, any user can submit a job using the XX job description, and be authorized to whatever JONES is authorized to. To avoid any security exposure, *do not* authorize this type of job description to *PUBLIC. Also, if this type of job description is used on a work station entry, it allows anyone to sign on as that user just by pressing the Enter key.

Every job in the system uses a job description during job initiation. This controls the various attributes of a job. The USER parameter controls the name of the user profile that will be assigned to the job. There are several considerations to this parameter.

Interactive Jobs: The job description to be used is defined on the ADDWSE (Add Work Station Entry) command. The default is to use the job description specified in the user profile. If USER (*RQD) is specified in the job description, the user must enter a user name. If USER(xxxx) is specified (where xxxx is a specific user profile name), the user is allowed to press the Enter key on the sign-on display and operate under the xxxx user profile name, unless security level 40 is in effect.

Batch Jobs: The job description used for batch jobs is specified on the Submit Job (SBMJOB) or Batch Job (BCHJOB) command.

If an input stream is entered that contains the BCHJOB command, the user entering the STRxxxRDR or SBMxxxJOB command must have object operational authority to the job description specified. When an input stream is used, jobs always operate under the user profile of the job description and not of the user who is placing the jobs on the job queue. If USER(*RQD) is specified in the job description, it is invalid to use the job description on a BCHJOB command.

If a SBMJOB command is used, the command defaults so that the batch job operates under the user profile name of the submitter. However, if USER(*JOBD) is specified on the SBMJOB command, the job operates under the name in the job description. The submitter can specify USER(*JOBD) only if they have *CHANGE authority to the job description. (No authority is needed to the actual user profile, unless security level 40 is in effect.)

Frequently, a specific name in the job description is required to let users submit work for a specific user profile. For example, the QBATCH job description is shipped with USER(QPGMR) to allow this. This job description is created normally (the public is given *CHANGE authority). This means that any user on the system who has authority to the SBMJOB, STRxxxRDR, or SBMxxxJOB command can submit work under the QPGMR user profile. You may want to change the QBATCH job description depending on your security needs.

Note: The IBM-supplied subsystem descriptions have been provided as examples and as backup for user-created subsystem descriptions. Therefore, we do not recommend modifying the subsystem descriptions in libraries QSYS and QGPL. You should make copies of the subsystem descriptions in these libraries and make changes to the copies.

Run-Time Attributes (Class Object)

A class object contains parameters that control the run-time environment of a job. Some of the parameters that are important to work management are:

Machine run priority (RUNPTY)
: The priority (1 through 99, with 1 as the highest priority) of a job when competing with other jobs for machine resources.

Purge (PURGE)
: Indicates if the working storage for a routing step is to be removed from main storage and placed in auxiliary storage at the end of a time slice or when entering a long wait. See Chapter 13, "Performance Tuning," for more information about the PURGE parameter.

Time slice (TIMESLICE)
The quantity of processor time (specified in milliseconds) allowed for a job before other waiting jobs are given the opportunity to run. See Chapter 13, "Performance Tuning," for more information about the TIMESLICE parameter.

Default maximum wait time (DFTWAIT)
Specifies the default maximum wait time (in seconds) that processing of a job is to be held up until the AS/400 instruction that performs a wait completes its running, such as a lock instruction.

Maximum processing time (CPUTIME)
The maximum amount of processor time (specified in milliseconds) a job can run before it is automatically ended by the system.

Maximum temporary storage (MAXTMPSTG)
The maximum amount of temporary auxiliary storage (specified in KB) a job can use before it is ended by the system.

Note: Maximum processing unit time and maximum temporary storage limits can help prevent an erroneous program from impairing system performance. However, a sufficient amount of processing time and temporary storage must be given to allow the job to complete its function.

The IBM-supplied classes are designed to meet the needs of typical interactive and batch applications. You can create your own classes. For example, you can create new classes to give you a wider range of machine running priorities.

The priorities for jobs that run in batch environments should normally be lower than priorities for jobs in interactive environments. You may want the priority for the system operator's jobs to be higher than priorities of other jobs so that the system operator can effectively respond to system needs. Also, the time slice should be small enough so that a looping program does not dominate processor time and an activity level.

If you use QCTL as the controlling subsystem, the operator is automatically running at a higher priority after signing on at the console. This is because QCTL routes the console job using the QCTL class, which specifies a higher priority. Another way you could set up your system so that the operator would run at a higher priority would be to do the following:

1. When the subsystem that the operator would be running in is not active, add a routing entry to it with unique routing data and specify the QSYS/QCTL class.
2. Create a new job description for the operator, specifying the same unique routing data that you used on the routing entry.
3. Change the operator's user profile to specify the new job description.
4. Start the subsystem.
5. Now when the operator signs on to that subsystem, the job will route using the QCTL class, which specifies a higher priority than the class used by normal interactive jobs.

Rerouting and Transferring Jobs

Rerouting a job (RRTJOB command) starts a new routing step in the same subsystem; transferring a job (TFRJOB or TFRBCHJOB commands) starts a new routing step in the same or another subsystem. In both cases, file overrides are removed, allocated objects except the work station and job message queue are deallocated, and files are closed. The library list stays the same because it is specified at the job level. The temporary library (QTEMP) and all objects in it remain the same, and all job attributes remain the same. Prestart jobs cannot be rerouted or transferred.

An interactive job is rerouted when either of the following occurs:

- The work station user presses the Exit key or enters the RETURN command from the only program in the program stack of a job. If QCMD is the only program on the program stack, a window is displayed with choices Y (=Yes) or N (=No) for exiting the command entry display.

  If the user responds with Y, the job is rerouted. If the user responds with N, the user can return to the command entry display.
- The work station user enters either the RRTJOB command or the TFRJOB command.

Rerouting a job can be used to change the processing attributes of a job. By rerouting a job, you

can use a different routing entry to process the next routing step. For example, if the next part of a job needs to run in a different storage pool, you reroute the job so that it runs in a different storage pool.

If an error is detected during routing step starting, the routing step is started only to produce a job log and perform clean-up functions. The routing step started message is still sent and gives the user the name of the subsystem in which the routing step failure occurred.

Either an interactive job or a batch job can be transferred to another subsystem. However, to transfer a job, the user profile for the job must have *READ authority for the job queue and *USE authority for the subsystem description of the subsystem being transferred to.

A transferring interactive job enters the new subsystem through the job queue. To transfer an interactive job, the job queue being transferred to (representing the subsystem being transferred to) must belong to an active subsystem and there must be a work station entry for the work station in the subsystem description of the subsystem being transferred to. (The work station entry must be AT(*ENTER) and not AT(*SIGNON).) When an interactive job is placed on a job queue, it is given the highest job queue priority to minimize any delay. However, if the job queue is being held or if the maximum number of active jobs is already met for the subsystem, the transferring job must wait until the queue is released or until an active job ends. If the new subsystem is ended by the End Subsystem (ENDSBS), End System (ENDSYS), or Power Down System (PWRDWNSYS) command while a transferring interactive job is on a job queue, the job is ended as part of the subsystem ending.

A batch job can transfer itself to another job queue. In this case, the job queue need not be associated with an active subsystem, and it is placed on the job queue with its current job priority.

Note: If a batch job is on a job queue, it can be moved to a different job queue by specifying the JOBQ parameter on the CHGJOB command.

Spooled inline files can be accessed only during the first routing step of a batch job. If a batch job contains a Transfer Job (TFRJOB), Reroute Job (RRTJOB), or Transfer Batch Job (TFRBCHJOB) command the spooled inline files cannot be accessed in the new routing step.

Because a PWRDWNSYS command prevents new jobs or routing steps from being started by any subsystem, a batch job that has been placed on a job queue by a TFRJOB command may not complete before the system is powered down. The temporary objects associated with a transferring job (such as the library list and the QTEMP library and all objects in it) are cleared during a power down; therefore, during the initial program load (IPL), the system is unable to restore the job to its previous state. During the IPL, the system removes the job from the job queue and produces its job log.

However, if a batch job has been placed on a job queue by the Transfer Batch Job (TFRBCHJOB) command, the system saves the information that is necessary to start the batch job again if it is on the job queue during an IPL. This includes having the same job name and a single job log made for the job. The temporary library (QTEMP) is cleared by the TFRBCHJOB command at each routing step.

The routing program is often used only once per job. If it calls another program and that program eventually does a return to the routing program, the following occurs:

- If QCMD is the routing program, the initial program is called (if any), and the initial menu is shown unless *SIGNOFF is specified in the INLMNU keyword of the user profile.
- If a call is from a user-written routing program, the return occurs as normal to the next instruction.
- If a user-written routing program does a TFRCTL, when the return occurs the routing program is called again.

The *CL Programmer's Guide* discusses how to write a program.

Job Logs

A job log contains information related to requests entered for a job. A special type of log, a history log (QHST), contains system data, such as a history of job activity on your system.

Logging Messages in the Job Log

Each job has an associated job log that can contain the following for the job:

- The commands in the job
- The commands in a CL program if the CL program was created with the LOG(*YES) option or with the LOG(*JOB) option and a Change Job (CHGJOB) command was run with the LOGCLPGM(*YES) option
- All messages (the message and help text for the message) sent to the requester and not removed from the program message queues

At the end of the job, the job log can be written to the spooled file QPJOBLOG so that it can be printed. After the job log is written to the spooled file, the job log is deleted. You can prevent a job log from being written for successfully completing jobs. See the discussion on preventing job logs later in this chapter.

You can control what information is written in the job log. To do this, you specify the LOG parameter on the Create Job Description (CRTJOBD) command. You can change these levels using the Change Job (CHGJOB) command or the Change Job Description (CHGJOBD) command. The LOG parameter is made up of 3 values: message level, message severity, and message text level. The first value, message level, has the following levels:

Level **Description**

0 No data is logged.

1 The only information to be logged is all messages sent to the job's external message queue with a severity greater than or equal to the message severity specified. Messages of this type indicate when the job started, when it ended, and its status at completion.

2 The following information is logged:

- Logging level 1 information
- Any requests or commands being logged from a CL program which receives messages on its program message queue with a severity greater than or equal to the severity specified
- All messages associated with any request or commands being logged from a CL program which receives messages issued with a severity greater than or equal to the severity specified

3 The following information is logged:

- Logging level 1 information
- All requests or commands being logged from a CL program
- All messages associated with any request or commands being logged from a CL program which receives messages issued with a severity greater than or equal to the severity specified

4 The following information is logged:

- All requests or commands being logged from a CL program
- All messages with a severity code greater than or equal to the severity specified

The second value, message severity, specifies the minimum severity level that causes error messages to be entered in the job log. Values 0 through 99 are allowed. Only those messages with a severity greater than or equal to this value are entered in the job log.

The third value in the LOG parameter, message text level, specifies the level of message text that is written in the job log. The values are:

***MSG** Only the message text is written to the job log (no message help text is included).

***SECLVL** Both the message and help text of the error message are logged in the job's log.

***NOLIST** No job log is produced if the job ends normally. If the job end code is 20 or greater, a job log is produced with the message and help text included.

| **Note:** Setting the log level to 0 0 *NOLIST in a lengthy batch job can waste system resource. This can occur when filtering only occurs at the end of the job and if 0 0 *NOLIST was specified

for the logging level. In this case, every message will be removed.

Filtering is the process of removing messages from the job log based on the message logging level of the job. If you run a CL program as an interactive or batch job, filtering runs once after the program ends. If you run a request processing program, filtering occurs before each new request is received. (Details on request processing programs can be found in the *CL Programmer's Guide*.)

The following example shows that filtering occurs after each command is run interactively. This example also shows how the logging level affects the information that is stored in the job message queue and, therefore, written to the job log, if one is requested:

First step: The CHGJOB command specifies a logging level of 2 and a message severity of 50, and that only messages are to be written to the job log (*MSG).

```
                              Command Entry                    SYSTEM1
                                                    Request level:   1
Previous commands and messages:

     > CHGJOB LOG(2 50 *MSG)
```

Second step: PGMA sends three informational messages with severity codes of 20, 50, and 60 to its own program message queue and to the program message queue of the program called before the specified call (*PRV). The messages that PGMA sends to its own program message queue are called **detailed messages**. These messages are are sent to the program message queue of the lower level program call.

PGMB sends two informational messages with severity codes of 40 and 50 to its own program message queue. These are detailed messages. PGMB also sends one informational message with a severity code of 10 to *PRV.

Note that the CHGJOB command no longer appears on the display. According to logging level 2, only requests for which a message has been issued with a severity equal to or greater than that specified are saved for the job log, and no messages were issued for this request. If such a message had been issued, any detailed messages that had been issued would also appear.

```
                              Command Entry                    SYSTEM1
                                                    Request level:   1
Previous commands and messages:

     > CALL PGMA
       Message sev 20 - PGMA
       Message sev 50 - PGMA
       Message sev 60 - PGMA
     > CALL PGMB
       Message sev 10 - PGMB




                                                                Bottom
Type command, press Enter.
===> _____________________________________________________________________
_____________________________________________________________________
_____________________________________________________________________
F3=Exit   F4=Prompt   F9=Retrieve   F10=Include detailed messages
F11=Display full   F12=Cancel   F13=Information Assistant F24=More keys
```

Third step: When F10=Include detailed messages is pressed from the Command Entry display, all the messages are displayed.

```
                              Command Entry                    SYSTEM1
                                                    Request level:   1
Previous commands and messages:

   > CALL PGMA
     Detailed message sev  20 - PGMA
     Detailed message sev  50 - PGMA
     Detailed message sev  60 - PGMA
     Message sev 20 - PGMA
     Message sev 50 - PGMA
     Message sev 60 - PGMA
   > CALL PGMB
     Detailed message sev  40 - PGMB
     Detailed message sev  50 - PGMB
     Message sev 10 - PGMB
                                                                Bottom
Type command, press Enter.
===> _____________________________________________________________________
_____________________________________________________________________
_____________________________________________________________________
F3=Exit   F4=Prompt   F9=Retrieve   F10=Include detailed messages
F11=Display full  F12=Cancel  F13=Information Assistant  F24=More keys
```

Fourth step: When another command is entered (in this example, CHGJOB), the `CALL PGMB` command and all messages (including detailed messages) are removed because the severity code for the message associated with this request was not equal to or greater than the severity code specified in the CHGJOB command. The `CALL PGMA` command and its associated messages remain because at least one of the messages issued for that request has a severity code equal to or greater than that specified.

On the following display, the CHGJOB command specifies a logging level of 3, a message severity of 40, and that both the first- and second-level text of a message are to be written to the job log. When another command is entered, the CHGJOB command remains on the display because logging level 3 logs all requests.

PGMC sends two messages with severity codes of 30 and 40 to the program message queue of the program called before the specified call (*PRV).

PGMD sends a message with a severity of 10 to *PRV.

```
                                Command Entry                    SYSTEM1
                                                    Request level:   1
Previous commands and messages:

     > CALL PGMA
       Message sev 20 - PGMA
       Message sev 50 - PGMA
       Message sev 60 - PGMA
     > CHGJOB LOG(3 40 *SECLVL)
     > CALL PGMC
       Message sev 30 - PGMC
       Message sev 40 - PGMC
     > CALL PGMD
       Message sev 10 - PGMD

                                                                   Bottom
Type command, press Enter.
===> ____________________________________________________________________
____________________________________________________________________
____________________________________________________________________
____________________________________________________________________
 F3=Exit   F4=Prompt   F9=Retrieve   F10=Include detailed messages
 F11=Display full   F12=Cancel   F13=Information Assistant   F24=More keys
```

When another command is entered after the CALL PGMD command was entered, the CALL PGMD command remains on the display, but its associated message is deleted. The message is deleted because its severity code is not equal to or greater than the severity code specified on the LOG parameter of the CHGJOB command.

The command SIGNOFF *LIST is entered to print the job log.

The job log, which follows, contains all the requests and all the messages that have remained on the Command Entry display and on the detailed messages display. In addition, the job log contains the second-level message text associated with each message, as specified by the CHGJOB command. Notice that the job log contains the second-level message text of any message issued during the job, not just for the messages issued since the second CHGJOB command was entered.

```
                                Command Entry                    SYSTEM1
                                                    Request level:   1
Previous commands and messages:

     > CHGJOB LOG(3 40 *SECLVL)
     > CALL PGMC
       Message sev 30 - PGMC
       Message sev 40 - PGMC
     > CALL PGMD
     > CALL PGME
     > SIGNOFF *LIST



                                                                   Bottom
Type command, press Enter.
===> ____________________________________________________________________
____________________________________________________________________
____________________________________________________________________
____________________________________________________________________
 F3=Exit   F4=Prompt   F9=Retrieve   F10=Include detailed messages
 F11=Display full   F12=Cancel   F13=Information Assistant   F24=More keys
```

Figure 4-3 on page 4-13 shows a job log. The headings at the top of each page of the printed job log identify the job to which the job log applies and the characteristics of each entry:

- The fully qualified name of the job (job name, user name, and job number).
- The name of the job description used to start the job.
- The date and time the job started.
- The message identifier.
- The message type.
- The message severity.
- The date and time each message was sent.
- The message. If the logging level specifies that second-level text is to be included, the second-level text appears on subsequent lines below the message.
- The program from which the message or request was sent.
- The machine interface instruction number or the offset into the program from which the message was sent.
- The program to which the message or request was sent.
- The machine interface instruction number or the offset into the program to which the message was sent.

  Note: The machine interface instruction numbers appear only for escape, notify, and diagnostic messages. For all other message types, the machine interface instruction number is set to zero.

The logging levels affect a batch job log in the same way as shown in the example in Figure 4-3 on page 4-13.

If the job uses APPC, the heading contains a line showing the unit of work identifier for APPC. See the *APPC Programmer's Guide* for more information on APPC.

Displaying the Job Log: The way to display a job log depends on the status of the job.

- If the job has ended and the job log is not yet printed, find the job using the Work with User Job (WRKUSRJOB) command, then:
  - Select option 8 (Display spooled file)
  - Find the spooled file named QPJOBLOG
  - Select option 5 (Display job log)
- If the job is still active (batch or interactive jobs) or is on a job queue and has not yet started, use the Work with User Job (WRKUSRJOB) command or the Work with Active Job (WRKACTJOB) command to display the job log as follows:
  - Select option 5 (Work with job)
  - Select option 10 (Display job log)

To display the job log of your own interactive job, do one of the following:

- Enter the following command:

  ```
  DSPJOBLOG
  ```

- Enter the WRKJOB command and select option 10 (Display job log) from the Work with Job display.
- Press F10 (Display detailed messages) from the Command Entry display (this key displays the messages that are shown in the job log).
- If the input inhibited light on your display station is on and remains on, do the following:
  1. Press the System Request key, then press the Enter key.
  2. On the System Request menu, select option 3 (Display current job).
  3. On the Display Job menu, select option 10 (Display Job log, if active or on job queue).
  4. On the Job Log display, press F10 (Display detailed messages).
  5. On the Display All Messages display, press the Roll Down key to see messages that were received before you pressed the System Request key.
- Sign off the work station, specifying LOG(*LIST) on the SIGNOFF command.

When you use the Display Job Log (DSPJOBLOG) command, you see the Job Log display. This display shows program names with special symbols, as follows:

>>	The running command or the next command to be run. For example, if a CL or high-level language program was called, the call to the program is shown.
>	The command has completed processing.
. .	The command has not yet been processed.
?	Reply message. This symbol marks both those messages needing a reply and those that have been answered.

On the Job Log display, you can do the following:

- Press F10 to display detailed messages. This display shows the commands or operations that were run within a high-level language program or within a CL program for which LOGCLPGM is activated.
- Use the Page down (or Roll up) key to get to the end of the job log. To get to the end of the job log quickly, press F18 (Bottom). After pressing F18, you might need to roll down to see the command that is running.
- Use the Page up (or Roll down) key to get to the top of the job log. To get to the top of the job log quickly, press F17 (Top).

Job Log Considerations: The following suggestions apply to using job logs:

- To change the output queue for all jobs on the system, use the OUTQ or DEV parameter on the Change Printer File (CHGPRTF) command to change the file QSYS/QPJOBLOG. The following are two examples using each of the parameters:

  ```
  CHGPRTF FILE(QSYS/QPJOBLOG) DEV(USRPRT)

  and

  CHGPRTF FILE(QSYS/QPJOBLOG) OUTQ(USROUTQ)
  ```

```
 5738SS1 V2R3M0 920925                          Job Log                          RCHAS727 12/12/92 07:58:53          Page    1
  Job name . . . . . :   QPADEV0007      User . . . . . :   SIMPSON      Number . . . . :   004201
  Job description  . . . . . :   QDFTJOBD        Library . . . . . :   QGPL
MSGID      TYPE                  SEV   DATE       TIME       FROM PGM       LIBRARY     INST     TO PGM       LIBRARY      INST
CPF1124    Information             00    12/12/92   07:57:16   QWTPIIPP       QSYS        04FC     *EXT                      0000
             Message . . . . :   Job 004201/SIMPSON/QPADEV0007 started on 12/12/92 at 07:57:16 in subsystem QINTER in QSYS. Job
               entered system on 12/12/92 at 07:57:16.
*NONE      Request                       12/12/92   07:57:50   QMHGSD         QSYS        0322     QCMD         QSYS         00B6
             Message . . . . :  -call pgma
CPF1001    Information             20    12/12/92   07:57:50   PGMA           SIMPSON     000C     PGMA         SIMPSON      000C
             Message . . . . :   Detailed message sev 20 - PGMA
             CPF1001 second level text - PGMA
CPF1002    Information             50    12/12/92   07:57:50   PGMA           SIMPSON     0010     PGMA         SIMPSON      0010
             Message . . . . :   Detailed message sev 50 - PGMA
             CPF1002 second level text - PGMA
CPF1003    Information             60    12/12/92   07:57:50   PGMA           SIMPSON     0014     PGMA         SIMPSON      0014
             Message . . . . :   Detailed message sev 60 - PGMA
             CPF1003 second level text - PGMA
CPF1004    Information             20    12/12/92   07:57:50   PGMA           SIMPSON     0018     QCMD         QSYS         00DE
             Message . . . . :   Message sev 20 - PGMA
             CPF1004 second level text - PGMA
CPF1005    Information             50    12/12/92   07:57:50   PGMA           SIMPSON     001C     QCMD         QSYS         00DE
             Message . . . . :   Message sev 50 - PGMA
             CPF1005 second level text - PGMA
CPF1006    Information             60    12/12/92   07:57:50   PGMA           SIMPSON     0020     QCMD         QSYS         00DE
             Message . . . . :   Message sev 60 - PGMA
             CPF1006 second level text - PGMA
*NONE      Request                       12/12/92   07:58:31   QMHGSD         QSYS        0322     QCMD         QSYS         00B6
             Message . . . . :  -chgjob log(3 40 *seclvl)
*NONE      Request                       12/12/92   07:58:34   QMHGSD         QSYS        0322     QCMD         QSYS         00B6
             Message . . . . :  -call pgmc
CPF100F    Information             30    12/12/92   07:58:34   PGMC           SIMPSON     000C     QCMD         QSYS         00DE
             Message . . . . :   Message sev 30 - PGMC
             CPF100F second level text - PGMC
CPF1010    Information             40    12/12/92   07:58:34   PGMC           SIMPSON     0010     QCMD         QSYS         00DE
             Message . . . . :   Message sev 40 - PGMC
             CPF1010 second level text - PGMC
*NONE      Request                       12/12/92   07:58:38   QMHGSD         QSYS        0322     QCMD         QSYS         00B6
             Message . . . . :  -call pgmd
*NONE      Request                       12/12/92   07:58:45   QMHGSD         QSYS        0322     QCMD         QSYS         00B6
             Message . . . . :  -call pgme
*NONE      Request                       12/12/92   07:58:52   QMHGSD         QSYS        0322     QCMD         QSYS         00B6
             Message . . . . :  -signoff *list
CPF1164    Completion              00    12/12/92   07:58:52   QWTMCEOJ       QSYS        01EE     *EXT                      0000
             Message . . . . :   Job 004201/SIMPSON/QPADEV0007 ended on 12/12/92 at 07:58:52; 3 seconds used; end code 0 .
             Cause . . . . . :   Job 004201/SIMPSON/QPADEV0007 completed on 12/12/92 at 07:58:52 after it used 3 seconds processing
               unit time.  The job had ending code 0. The job ended after 1 routing steps with a secondary ending code of 0.  The
               job ending codes and their meanings are as follows:  0 - The job completed normally. 10 - The job completed normally
               during controlled ending or controlled subsystem ending. 20 - The job exceeded end severity (ENDSEV job attribute).
               30 - The job ended abnormally. 40 - The job ended before becoming active. 50 - The job ended while the job was
               active. 60 - The subsystem ended abnormally while the job was active. 70 - The system ended abnormally while the job
               was active. 80 - The job ended (ENDJOBABN command). 90 - The job was forced to end after the time limit ended
               (ENDJOBABN command). Recovery  . . . :   For more information, see the Work Management Guide, SC41-8078.
```

Figure 4-3. Job Log

- To specify the output queue to which a job's job log is written, make sure that file QPJOBLOG has OUTQ(*JOB) specified. You can then use the OUTQ parameter on any of the following commands: BCHJOB, CRTJOBD, CHGJOBD, or CHGJOB. The following is an example:

  ```
  CHGJOB OUTQ(USROUTQ)
  ```

- If the output queue for a job cannot be found, no job log is produced.
- To hold all job logs, specify HOLD(*YES) on the CHGPRTF command for the file QSYS/QPJOBLOG. The job logs are then released to the writer when the Release Spooled File (RLSSPLF) command is run. The following is an example:

  ```
  CHGPRTF FILE(QSYS/QPJOBLOG) HOLD(*YES)
  ```

- If the system abnormally ends, the start prompt allows the system operator to specify whether the job logs are to be printed for any jobs that were active at the time of the abnormal end.
- To delete a job log, use the Delete Spooled File (DLTSPLF) command or the Delete option on the output queue display.
- If you used the USRDTA parameter on the Change Print File (CHGPRTF) command to change the user data value for the QSYS/QPJOBLOG file, the value specified will not be shown on the Work with Output Queue or Work with All Spooled Files displays. The value shown in the user data column is the job name of the job whose job log has printed.

Considerations for Interactive Job Logs:

The IBM-supplied job descriptions QCTL, QINTER, and QPGMR all have a log level of LOG(4 0 *NOLIST); therefore, all messages help text are written to the job log. However, the job logs are not printed if the job ends normally unless you specify *LIST on the SIGNOFF command. To change the log level for interactive jobs, you can use the CHGJOB or CHGJOBD command.

If a display station user uses an IBM-supplied menu or the command entry display, all error messages are displayed. If the display station user uses a user-written initial program, any unmonitored message causes the initial program to end and a job log to be produced. However, if the initial program monitors for messages, it receives control when a message is received. In this case, it may be important to ensure that the job log is produced so you can determine the specific error that occurred. For example, assume that the initial program displays a menu that includes a sign-off option, which defaults to *NOLIST. The initial program monitors for all exceptions and includes a Change Variable (CHGVAR) command that changes the sign-off option to *LIST if an exception occurs:

```
         PGM
         DCLF MENU
         DCL &SIGNOFFOPT TYPE(*CHAR) LEN(7)
            VALUE(*NOLIST)
         .
         .
         .
         MONMSG MSG(CPF0000) EXEC(GOTO ERROR)
PROMPT:  SNDRCVF RCDFMT(PROMPT)
         CHGVAR &IN41 '0'
         .
         .
         .
         IF (&OPTION *EQ '90') SIGNOFF
            LOG(&SIGNOFFOPT)
         .
         .
         .
         GOTO PROMPT
ERROR:   CHGVAR &SIGNOFFOPT '*LIST'
         CHGVAR &IN41 '1'
         GOTO PROMPT
         ENDPGM
```

If an exception occurs, the CHGVAR command changes the option on the SIGNOFF command to *LIST and sets on an indicator. This indicator could be used to condition a constant that displays a message explaining that an unexpected error occurred and telling the display station user what to do.

If the interactive job is running a CL program, the CL program commands are logged only if the log level is 3 or 4 and one of the following is true:

- LOGCLPGM(*YES) was specified on the Create Control Language Program (CRTCLPGM) command.
- LOGCLPGM(*JOB) was specified on the Create Control Language Program (CRTCLPGM) command and (*YES) is the current LOGCLPGM job attribute.

The LOGCLPGM job attribute can be set and changed by using the LOGCLPGM parameter on the SBMJOB, CRTJOBD, and CHGJOBD commands.

Considerations for Batch Job Logs:

For your batch applications, you may want to change the amount of information logged. The log level (LOG(4 0 *NOLIST)) specified in the job description for the IBM-supplied subsystem QBATCH supplies a complete log if the job abnormally ends. If the job completes normally, no job log is produced.

If you want to print the job log in all cases, use the Change Job Description (CHGJOBD) command to change the job description, or specify a different LOG value on the BCHJOB or SBMJOB command.

If the batch job is running a CL program, the CL program commands are logged only if LOGCLPGM(*YES) is specified on the Create Control Language Program (CRTCLPGM) command or the Change Program (CHGPGM) command.

QHST History Log

The history log (QHST) consists of a message queue and a physical file known as a log-version. Messages sent to the log message queue are written by the system to the current log-version physical file.

- History log (QHST). Contains a high-level trace of system activities such as system, subsystem, and job information, device status,

and system operator messages. Its message queue is QHST.

When a log-version is full, a new version of the log is automatically created. Each version is a physical file that is named in the following way:

```
Qxxxyydddn
```

where:

xxx	Is a 3-character description of the log type (HST)
yyddd	Is the Julian date on which the log-version was created
n	Is a sequence number within the Julian date (0 through 9 or A through Z)

Note: The number of records in the version of the history log is specified in the system value QHSTLOGSIZ.

You can also write a program to process history log records. Because several versions of each log may be available, you must select the log-version to be processed. To determine the available log-versions, use the Display Object Description (DSPOBJD) command. For example, the following DSPOBJD command displays what versions of the history log are available:

```
DSPOBJD   OBJ(QSYS/QHST*)   OBJTYPE(*FILE)
```

You can delete logs on your system using the delete option from the display presented on the Work with Objects (WRKOBJ) command. You can display or print the information in a log using the Display Log (DSPLOG) command. You can select the information you want displayed or printed by specifying any combination of the following:

- Period of time
- Name of job that sent the log entry
- Message identifiers of entries

The following DSPLOG command displays all the available entries for the job OEDAILY in the current day:

```
DSPLOG JOB(OEDAILY)
```

The resulting display is:

```
                        Display History Log Contents

Job OEDAILY started
Database file OEMSTR in library OELIB expired
Job OEDAILY abnormally ended
Job OEDAILY started
Job OEDAILY ended

                                                                    Bottom
Press Enter to continue.

F3=Exit   F10=Display all   F12=Cancel
```

If you have reset the system date or time to an earlier setting, a system log-version can contain entries that are not in chronological order and, therefore, when you try to display the log-version, some entries may be missed. For example, if the log-version contains entries dated 1988 followed by entries dated 1987, and you want to display those 1987 entries, you specify the 1987 dates on the PERIOD parameter on the DSPLOG command but the expected entries are not displayed. You should always use the system date (QDATE) and the system time (QTIME), or you should specify the PERIOD parameter as follows:

```
PERIOD((start-time start-date) (*AVAIL *END))
```

The system writes the messages sent to a system log message queue to the current version physical file when the message queue is full or when the DSPLOG command was used. If you want to ensure the current version is up-to-date, specify a fictitious message identifier, such as `###0000`, on the DSPLOG command. No messages are displayed, but the log-version physical file is made current.

Format of the History Log

A database file is used to store the messages sent to a system log. Because all records in a physical file have the same length and messages sent to a log have different lengths, the messages can span more than one record. Each record for a message has three fields:

- System date and time (a character field of length 8). This is an internal field. The converted date and time also are in the message.
- Record number (a 2-byte field). For example, the field contains hex `0001` for the first record, hex `0002` for the second record, and so on.
- Data (a character field of length 132).

Figure 4-4 on page 4-16 shows the format for the third field (data) of the first record.

Figure 4-4. Format for Third Field of the First Record

Contents	Type	Length	Positions in Record
Job name	Character	26	11-36
Converted date and time[1]	Character	13	37-49
Message ID	Character	7	50-56
Message file name	Character	10	57-66
Library name	Character	10	67-76
Message type[2]	Character	2	77-78
Severity code	Character	2	79-80
Sending program name	Character	12	81-92
Sending program instruction number	Character	4	93-96
Receiving program name	Character	10	97-106
Receiving program instruction number	Character	4	107-110
Message text length	Binary	2	111-112
Message data length	Binary	2	113-114

Figure 4-4. Format for Third Field of the First Record

Contents	Type	Length	Positions in Record
Reserved	Character	28	115-142

1 The format is: `cyymmddhhmmss`
where:

`c`	Is the century digit (c=0 if yy ≥ 40, c = 1 if yy < 40)
`yymmdd`	Is the year, month, and day that the message is sent
`hhmmss`	Is the hour, minute, and second that the message is sent

2 This has the same value as the RTNTYPE parameter on the Receive Message (RCVMSG) command.

Figure 4-5 shows the format of the third field (data) of the remaining records.

Figure 4-5. Format of Remaining Records

Contents	Type	Length
Message	Character	Variable[1]
Message data	Character	Variable[2]

1 This length is specified in the first record (positions 111 and 112) and cannot exceed 132.

2 This length is specified in the first record (positions 113 and 114).

A message is never split when a new version of a log is started. The first and last records of a message are always in the same QHST version.

Processing the QHST File

If you use a high-level language program to process the QHST file, keep in mind that the length of the message can vary with each occurrence of a message. Because the message includes replaceable variables, the actual length of the message varies; therefore, the message data begins at a variable location for each use of the same message.

However, for message CPF1124 (job start) and message CPF1164 (job completion), the message data always begins in position 11 of the third record.

Performance information is not displayed as text on message CPF1164. Because the message is in the QHST log, users can write application programs to retrieve this data. The format of this performance information is as follows.

The performance information is passed as a variable length replacement text value. This means that the data is in a structure with the first entry being the length of the data. The size of the length field is not included in the length. The first data fields in the structure are the times and dates that the job entered the system and when the first routing step for the job was started. The times are in the format 'hh:mm:ss'. The time separators in this example are colons. This separator is determined by the value specified in the QTIMSEP system value. The dates are in the format defined in the system value QDATFMT and the separators are in the system value QDATSEP. The time and date the job entered the system precede the job start time and date in the structure.

The time and date the job entered the system are when the system becomes aware of a job to be initiated (a job structure is set aside for the job). For an interactive job, the job entry time is the time the password is recognized by the system. For a batch job, it is the time the BCHJOB or SBMJOB command is processed. For a monitor job, reader or writer, it is the time the corresponding start command is processed, and for autostart jobs it is during the start of the subsystem.

Following the times and dates are the total response time and the number of transactions. The total response time is in seconds and contains the accumulated value of all the intervals the job was processing between pressing the Enter key at the work station and when the next display is shown. This information is similar to that shown on the WRKACTJOB display. This field is only meaningful for interactive jobs.

It is also possible in the case of a system failure or abnormal job end that the last transaction will not be included in the total. The job end code in this case would be a 40 or greater. The transaction count is also only meaningful for interactive jobs other than the console job and is the number of response time intervals counted by the system during the job.

The number of synchronous auxiliary I/O operations follows the number of transactions. This is the same as the AUXIO field that appears on the WRKACTJOB display except that this value is the total for the job. If the job ends with a end code of 70, this value may not contain the count for the final routing step. Additionally, if a job exists across an IPL (using a TFRBCHJOB command) it is ended before becoming active following an IPL, the value is 0.

The final field in the performance statistics is the job type. Values for this field are:

A Autostarted job
B Batch job
I Interactive job
M Subsystem monitor
R Spooling reader
S System job
W Spooling writer
X Start job

For messages in which the message data begins in a variable position, you can access the message data by doing the following:

- Determine the length of the variables in the message. For example, assume that a message uses the following five variables:

```
Job name     *CHAR 10
User name    *CHAR 10
Job number   *CHAR 6
Time         *CHAR 8
Date         *CHAR 8
```

  These variables are fixed in the first 42 positions of the message data.

- To find the location of the message data, consider that:
  - The message always begins in position 11 of the second record.
  - The length of the message is stored in a 2-position field beginning at position 111 of the first record. This length is stored in a binary value so if the message length is 60, the field contains hex 003C.

  Then, by using the length of the message and the start position of the message, you can determine the location of the message data.

Reason Codes for Message CPF1164:

If no job log is produced for a job, you can use the following reason codes for CPF1164 to locate information in the QHST file that more precisely describes why a job ended. Figure 4-6 on page 4-18 lists and describes the reason codes.

Figure 4-6. Reason Codes for CPF1164

Reason Codes	Description
0	No special reason for job ending.
100	The disconnect time interval was exceeded.
101	The session device was deleted and created again for a disconnected job.
102	Errors occurred trying to start a job that was disconnected.
300	Job encountered a device error and DEVRCYACN was set to *ENDJOB.
301	Job ended due to looping on device errors.

Example of Processing the QHST File:

The following RPG/400* program processes the QHST file for the job completion message CPF1164. The QHST file in the program must be overridden to one of the actual QHST files. The program reads each record from the file into an array. A data structure is defined to match the message data for CPF1164. You can determine the structure of the message data for a message by using the Display Message Description (DSPMSGD) command.

For message CPF1164, there are always three records and the message data always begins in position 11 of the third record. The program checks for the first record of a message (RCDNBR = 1) and then checks for CPF1164 in positions 50 through 56. If the message is CPF1164, the program sets on indicator 22 so that when the third record is read, the message data can be moved to the data structure. Once the message data is in the data structure, the fields can be printed, as in this example, or written to a database file. Note that the RTGSTP field contains the number of routing steps used by the job. Some of the information in the message data is not in the text for CPF1164. This includes:

- Number of routing steps
- Time and date the job was entered (this is the submitted time for batch jobs; for interactive jobs, it is the same as the time and date started)
- Time and date started
- Total response time (*)
- Number of transactions (*) (number of display station interactions)
- Number of auxiliary storage I/O operations (*)
- Job type

Note: The asterisked (*) fields are the same as those on the WRKACTJOB display. Figure 4-7 on page 4-19 is a programming example for processing the QHST file.

```
SEQNBR *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ..

 01.00      FQHST    IP  F     142            DISK
 02.00      FQPRINT  O   F     132     OF     PRINTER
 03.00      E                    ARR       132  1              INPUT ARRAY
 04.00      IQHST    AA  01
 05.00      I                                      B   9  100RCDNBR
 06.00      I                                         11 142 ARR
 07.00      I                                         50  56 MSGNBR
 08.00      IMSG         DS
 09.00      I                                          1  10 JOB
 10.00      I                                         11  20 USER
 11.00      I                                         21  26 JOBNBR
 12.00      I                                         27  34 JOBTIM
 13.00      I                                         35  42 JOBDAT
 14.00      I                                      B  43  460CPUTIM
 15.00      I                                      B  47  480RTGSTP
 16.00      I                                      B  49  500CMPCOD
 17.00      I                                         73  80 TIMENT
 18.00      I                                         81  88 DATENT
 19.00      I                                         89  96 STRTIM
 20.00      I                                         97 104 STRDAT
 21.00      I                                      B 105 1080TOTRSP
 22.00      I                                      B 109 1120NBRTRN
 23.00      I                                      B 113 1160NBRAUX
 24.00      I                                        117 117 JOBTYP
 25.00      C           1         COMP RCDNBR                  20
 26.00      C   20      'CPF1164' COMP MSGNBR                  22
 27.00      C   20                GOTO END
 28.00      C           3         CABNERCDNBR    END           12
 29.00      C  N22                GOTO END
 30.00      C                     MOVEAARR       MSG
 31.00      C                     EXCPTDETAIL
 32.00      C           END       TAG
 33.00      OQPRINT  E  1             DETAIL
 34.00      O                         JOB
 35.00      O                         USER
 36.00      O                         JOBNBR
 37.00      O                         JOBTIM  +  1
 38.00      O                         JOBDAT  +  1
 39.00      O                         CPUTIM  +  1
 40.00      O                         RTGSTP  +  1
 41.00      O                         CMPCOD  +  1
 42.00      O                         TIMENT  +  1
 43.00      O                         DATENT  +  1
 44.00      O                         STRTIM  +  1
 45.00      O                         STRDAT  +  1
 46.00      O                         TOTRSP  +  1
 47.00      O                         NBRTRN  +  1
 48.00      O                         NBRAUX  +  1
 49.00      O                         JOBTYP  +  1
```

Figure 4-7. QHST File Processing Example

How To's

This section tells you how to perform some common tasks involving job attributes and jobs.

Changing Job Attributes

The following examples allow you to change the attributes of a job:

- Temporarily for an interactive job

  ```
  CHGJOB LOG(4 0 *SECLVL) OUTQ(mylib/myoutq)
  ```

- For a single batch job

  ```
  SBMJOB CMD(xxxxxxxx)  LOG(4 0 *SECLVL)
  ```

- Permanently

  ```
  CHGJOBD JOBD(library/jobd) LOG(4 0 *SECLVL)
  ```

  The default job description used by user profiles is QGPL/QDFTJOBD. The default for submitting jobs is to use the job description from the user profile submitting the job.

Changing Priority and Time Slice

You can create a program that changes your priority and time slice for the duration of the command running and then resets the two values to what they were previously. For example, it may be desirable to have a special program that can be used in an emergency situation where the response time of command running is critical. To do this, you would enter the following program to improve the performance of the WRKACTJOB command:

```
PGM
DCL &RUNPTY *DEC LEN(2 0)
DCL &TIMESLICE  *DEC LEN(7 0)
RTVJOBA  RUNPTY(&RUNPTY) TIMESLICE(&TIMESLICE)
CHGJOB RUNPTY(1) TIMESLICE(20000)
WRKACTJOB
CHGJOB RUNPTY(&RUNPTY) TIMESLICE(&TIMESLICE)
ENDPGM
```

You may want to create a command for this function to allow a simple means of running this program.

If the WRKACTJOB command is being used to determine an immediate performance situation on the system, you should consider specifying the command as:

```
WRKACTJOB CPUPCTLMT(2) RESET(*YES)
```

When the display first appears, no active jobs are meeting the requirements because the active job statistics have been reset. If the Refresh key is pressed, the display shows all jobs that are using more than 2% of the processing unit resource since the command was run.

Finding a Job

To find a job on the system, use either the Work with Active Job (WRKACTJOB), Work with User Job (WRKUSRJOB), or Work with Submitted Job (WRKSBMJOB) command. After entering one of these commands, one of the Work with displays appears, showing you a list of jobs.

Determining Status of a Job

To determine the status of a job on the system, use either the Work with Active Job (WRKACTJOB) command or the Work with User Job (WRKUSRJOB) command. Use the WRKACTJOB command to determine the specific status of the job when the job is active, such as a message waiting. Use the WRKUSRJOB command to determine if the job is still on the job queue, is active in a subsystem, or is finished but with output still waiting to print. After entering one of these commands, one of the Work with displays appears, showing you a list of jobs. The status of these jobs is listed in the far right column.

Displaying Messages

To display the message a job is waiting for, use one of the following commands:

- Work with User Jobs (WRKUSRJOB)
- Work with Subsystem Jobs (WRKSBSJOB)
- Work with Active Jobs (WRKACTJOB)
- Work with Submitted Jobs (WRKSBMJOB)

After you press the Enter key, one of the Work with displays appears, showing you a list of jobs. If the job is waiting for a message, the status MSGW (message wait) is listed in the status column. Select option 7 (Display message) to display the message this job is waiting for.

Viewing a Job's Output

To view the output from a job, use either the Work with Active Job (WRKACTJOB) command, Work with User Job (WRKUSRJOB) command, or Work with Submitted Job (WRKSBMJOB) command to display a list of jobs. Select option 8 (Work with spooled files) to display the spooled files for that job. You can also use the Work with Jobs (WRKJOB) command with the value *SPLF specified for the Option parameter.

Ending Jobs

To end a job, use either the Work with Active Job (WRKACTJOB) command, Work with User Job (WRKUSRJOB) command, or Work with Submitted Job (WRKSBMJOB) command to display a list of jobs. Select option 4 (End) to end the job. If you know the name of the job you want to end, use the End Job (ENDJOB) command.

Changing Job Descriptions

To change a job description, first use the Work with Job Description (WRKJOBD) command to find the job description you want to change, then select Option 2 to make the change. If you know the name of the job description you want to change, use the Change Job Description (CHGJOBD) command. Enter CHGJOBD and select F4=Prompt. The Change Job Description display appears.

Changing Classes

Use the Work with Classes (WRKCLS) command to find the class and then select option 2 to change the class. You can also use the Change Class (CHGCLS) command if you know the name of the class you want to change.

Getting Job Logs in One Output Queue

The default output queue for job logs is the output queue the job is using.

Since the system looks at the print file first to determine which output queue to use, the job log print file output queue parameter can be changed to a specific output queue.

The following command changes the job log print file so that all job logs go to the specified output queue.

```
CHGPRTF FILE(QSYS/QPJOBLOG)
        OUTQ(library/outq)
```

Changing Logging Level for a Job

You can control the level of information that is put in the job log. To change the logging level:

- Temporarily for an interactive job

  ```
  CHGJOB LOG(4 0 *SECLVL)
  ```

- For a single batch job

  ```
  SBMJOB CMD(xxxxxxxx)  LOG(4 0 *SECLVL)
  ```

- Permanently

  ```
  CHGJOBD JOBD(library/jobd)
          LOG(4 0 *SECLVL)
  ```

  The default job description used by user profiles is QGPL/QDFTJOBD. The default for submitting jobs is to use the job description from the user profile submitting the job.

Help text for these commands describes the different logging levels.

Note: The examples above set the logging level to the highest logging level possible.

Preventing the Production of Job Logs

To prevent a job log from being produced at the completion of a batch job, you can specify *NOLIST for the message level text of the LOG parameter on the Batch Job (BCHJOB), Submit Job (SBMJOB), Change Job (CHGJOB), Create Job Description (CRTJOBD), or Change Job Description (CHGJOBD) command. If you specify *NOLIST for the message text value of the LOG parameter, the job log is not produced at the end of a job unless the job end code is 20 or greater. If the job end is 20 or greater, the job log is produced.

For an interactive job, the value specified for the LOG parameter on the SIGNOFF command takes precedence over the LOG parameter value specified for the job.

Deleting Log Files

Log-version physical files accumulate on a system and you should periodically delete old logs that are not needed. A log-version is created such that only the security officer is authorized to delete it.

To delete a log-version, use the Delete File (DLTF) command and specify the object name or generic name to be deleted.

Another alternative is to use the following sample command and program to simplify deleting logs on your system. The Delete Log (DLTLOG) command specifies the log and the number of days to keep logs. The default log is QHST, and the default number of days is five. For example, if you keep the defaults, all QHST files that are older than five days from the current date are deleted. The last date in which a message was sent to the log is used for the comparison, *not* the creation date of the log.

Because the security officer is the only user who can delete the log files, you may want to create the program with USRPRF(*OWNER) and authorize the command and program to specific users.

This function is dependent on the accurate entry of the system date during IPL You may want to add this command to other standard processing functions of QHST and include them as an autostart job.

The command accepts the name of the log to be deleted and the number of days that the logs are to be kept. The current date is retrieved and converted to Julian format. The program subtracts the number of days entered on the command from the last date in which a message was sent to the log, and that number is used. The text description of each log file contains the date that the last message was sent to it. The program also tests for year-end. The names of the log files are put into an output file in library QTEMP by the DSPOBJD command. The number of files deleted and kept is counted and placed in the completion message.

Enter the command definition statements and create the DLTLOG command with the parameter PGM(DLTLOGC) specified as shown in Figure 4-8.

```
SEQNBR *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ..

 01.00           CMD         PROMPT('Delete Log')  /* CPP IS DLTLOGC */
 02.00           PARM        LOG *CHAR LEN(4) DFT(QHST) RSTD(*YES) +
 03.00                          VALUES('QHST') +
 04.00                          PROMPT('Log (QHST):')
 05.00           PARM        DAYS *DEC LEN(2 0) DFT(5) +
 06.00                          RANGE(1 50) +
 07.00                          PROMPT('Nbr of days to keep log')
```

Figure 4-8. DLTLOG Example

Enter the CL program source statements and create the DLTLOGC program as shown in Figure 4-9 on page 4-23.

```
SEQNBR *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ..

 01.00            PGM         PARM(&LOG &DAYS)
 02.00            DCLF        QSYS/QADSPOBJ /* IBM file to be overridden */
 03.00            DCL         &LOG *CHAR LEN(4)
 04.00            DCL         &DAYS *DEC LEN(2 0)
 05.00            DCL         &WRKDAT *CHAR LEN(6)
 06.00            DCL         &JULIANA *CHAR LEN(5)
 07.00            DCL         &YRD *DEC LEN(2 0)
 08.00            DCL         &DAYSD *DEC LEN(3 0)
 09.00            DCL         &NUM3 *DEC LEN(3 0)
 00.00            DCL         &QHSTLAST *CHAR LEN(6)
 11.00            DCL         &QHSTLASTJ *CHAR LEN(5)
 12.00            DCL         &DLTCNT *DEC LEN(5 0)
 13.00            DCL         &DLTCNTA *CHAR LEN(5)
 14.00            DCL         &RMNCNT *DEC LEN(5 0)
 15.00            DCL         &RMNCNTA *CHAR LEN(5)
 16.00            DCL         &MSGID *CHAR LEN(7)
 17.00            DCL         &MSGDTA *CHAR LEN(100)
 18.00            MONMSG      MSGID(CPF0000) EXEC(GOTO ERROR)
 19.00                        /* Subtract nbr of days from the current date */
 20.00            RTVJOBA     DATE(&WRKDAT) /* Rtv current date */
 21.00            CVTDAT      DATE(&WRKDAT) TOVAR(&JULIANA) TOFMT(*JUL) +
 22.00                             TOSEP(*NONE) /* Convert to Julian */
 23.00            CHGVAR      &YRD %SST(&JULIANA 1 2)
 24.00            CHGVAR      &DAYSD %SST(&JULIANA 3 3)
 25.00            CHGVAR      &NUM3 (&DAYSD - &DAYS) /* Sub for cmd days */
 26.00            IF          (&NUM3 *LE 0) DO   /* Went passed Jan 1 */
 27.00            CHGVAR      &YRD (&YRD - 1)
 28.00            CHGVAR      &DAYSD (365 + &NUM3) /* Assume 365 per year */
 29.00            ENDDO
 30.00            ELSE        CHGVAR    &DAYSD &NUM3 /* Same year */
 31.00            CHGVAR      %SST(&JULIANA 1 2) &YRD  /* Cnvt to Julian */
 32.00            CHGVAR      %SST(&JULIANA 3 3) &DAYSD  /* Cnvt to Julian */
 33.00                        /* Create OUTFILE of log descriptions */
 34.00             DSPOBJD    OBJ(QSYS/(&LOG *CAT '*')) OBJTYPE(*FILE) +
 35.00                          OUTPUT(*OUTFILE) OUTFILE(QTEMP/DSPOUTP)
 36.00                        /* Delete log loop */
 37.00            OVRDBF      QADSPOBJ TOFILE(QTEMP/DSPOUTP)
 38.00 LOOP:      RCVF        RCDFMT(QLIDOBJD) /* DSPOBJD format */
 39.00            MONMSG      MSGID(CPF0864) EXEC(GOTO END)
 40.00                        /* Extract the date of the last msg sent */
 41.00                        /*   to the log file.  This is written   */
 42.00                        /*   as part of the text description of  */
 43.00                        /*   the object beginning in position 15 */
 44.00            CHGVAR      &QHSTLAST %SST(&ODOBTX 15 6)
 45.00            CVTDAT      DATE(&QHSTLAST) TOVAR(&QHSTLASTJ) +
 46.00                             FROMFMT(*YMD) TOFMT(*JUL) +
 47.00                             TOSEP(*NONE) /* Convert to Julian */
 48.00            IF          (&QHSTLASTJ *LE &JULIANA) DO /* Comp dates */
 49.00            DLTF        QSYS/&ODOBNM /* Delete the log file */
 50.00            CHGVAR      &DLTCNT (&DLTCNT + 1)  /* Count decisions */
 51.00            ENDDO
```

Figure 4-9 (Part 1 of 2). DLTLOGC CL Program Source Statements

```
SEQNBR *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ..
 52.00            ELSE        +
 53.00            CHGVAR      &RMNCNT (&RMNCNT + 1) /* Count remaining */
 54.00            GOTO        LOOP
 55.00                        /* Send completion message */
 56.00 END:                   CHGVAR &DLTCNTA &DLTCNT /* Convert to alpha */
 57.00            CHGVAR      &RMNCNTA &RMNCNT  /* Convert to alpha */
 58.00            SNDPGMMSG   MSG('Number of files deleted-' *CAT +
 59.00                              &DLTCNTA *CAT +
 60.00                              '    Number remaining-' *CAT +
 61.00                              &RMNCNTA)
 62.00            DLTF        QTEMP/DSPOUTP
 63.00            RETURN
 64.00 ERROR:     RCVMSG      MSGID(&MSGID) MSGDTA(&MSGDTA) MSGTYPE(*EXCP)
 65.00            SNDPGMMSG   MSGID(&MSGID) MSGDTA(&MSGDTA) +
 66.00                             MSGF(QCPFMSG) MSGTYPE(*ESCAPE)
 67.00            ENDPGM
```

Figure 4-9 (Part 2 of 2). DLTLOGC CL Program Source Statements

You could change this program to include the following functions:

- Copy the QHST file to another file.
- Save the QHST file.
- Produce standard reports on the file.

Chapter 5. Interactive Jobs

An interactive job starts when a user signs on to a display station and ends when the user signs off. For the job to run, the subsystem searches for the job description, which may be specified in the work station entry or the user profile.

Initiating Jobs

The Sign On display is an example of what the work station user sees at the work station. The use of the shipped objects by the OS/400 licensed program is not obvious to the work station user. For this example, the OS/400 licensed program and the subsystem QBASE have been started. It also assumes that the work station user has an initial main menu and no initial program is called that displays a menu.

```
                                Sign On
                                          System  . . . . . :   XXXXXXXX
                                          Subsystem . . . . :   QBASE
                                          Display . . . . . :   DSP01

          User  . . . . . . . . . . . . .   _______

          Program/procedure . . . . . . .   _______
          Menu  . . . . . . . . . . . . .   _______
          Current library . . . . . . . .   _______
```

Enter your user name and press the Enter key. The AS/400 Main Menu is shown.

```
MAIN                          AS/400 Main Menu
                                                          System   XXXXXXXX
Select one of the following:

     1. User tasks
     2. Office tasks
     3. General system tasks
     4. Files, libraries, and folders
     5. Programming
     6. Communications
     7. Define or change the system
     8. Problem handling
     9. Display a menu
    10. User support and education
    11. PC Support tasks

    90. Sign off

Selection or Command
===> _____________________________________________________________________
F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel   F13=User support
F23=Set initial menu  F13=Information Assistant  F24=More keys
```

The first display is the sign-on display. If the work station user enters a user name and presses the Enter key, the AS/400 Main Menu is displayed. The system default does not require a password.

The work station user can then select options from this menu. When option 90 (Sign off) is selected, the interactive job ends and the new sign-on display is shown.

You can change the QSECURITY system value to require that the user enters a valid password. See Chapter 2, "System Values and Network Attributes," or *Security Reference* for more information on requiring passwords.

Getting the Job's Attributes

Figure 5-1 shows what normally occurs when someone signs on to the system.

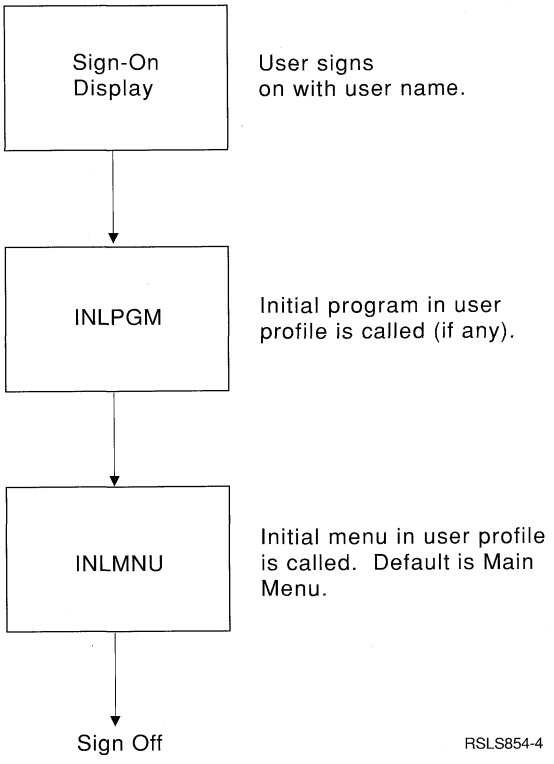


Figure 5-1. Programmer Signs On to the AS/400 System

The user types the user profile name and presses the Enter key. The Main Menu appears because the user does not have an initial program. The user can then specify commands or options from the menu to request functions of the system.

To make it easier to control and identify jobs on the system, each job has a unique qualified job

name. The qualified job name consists of three parts: the job name, the user name, and the job number.

- For interactive jobs, the job name is the same as the name of the work station you signed on to. For batch jobs you can specify your own job name. The job name can be up to 10 characters long.
- The user name is the name of the user profile under which the job is running. For interactive jobs, the user name is the name you entered in the user field on the sign-on display. For batch jobs you can specify the user profile under which the batch job is to run. The user name can be up to 10 characters long.
- The job number is assigned by the system so that you can always uniquely identify any job, even if more than one has the same job name and user name. The job number is always 6 numeric digits.

The syntax for qualified job names is similar to qualified names for objects. For example, if the job name is DSP01, the user is QPGMR, and the job number is 000578, the qualified job name would be entered on the Work with Job (WRKJOB) command as follows:

```
WRKJOB JOB(000578/QPGMR/DSP01)
```

Another similarity to object names is that you do not need to specify all of the qualifiers. For example you could specify the following:

```
WRKJOB JOB(QPGMR/DSP01)
```

or

```
WRKJOB JOB(DSP01)
```

This works the same as entering the entire qualified job name as long as there is only one job on the system that matches the portion of the qualified name that you entered. If several (more than one) jobs match the job you entered, the Select Job display appears. Use this display to choose the job you intended.

Routing the Job into the Subsystem

Figure 5-2 on page 5-3 shows the subsequent activity leading up to starting a routing step and displaying the Main Menu for a user profile specifying an initial program.

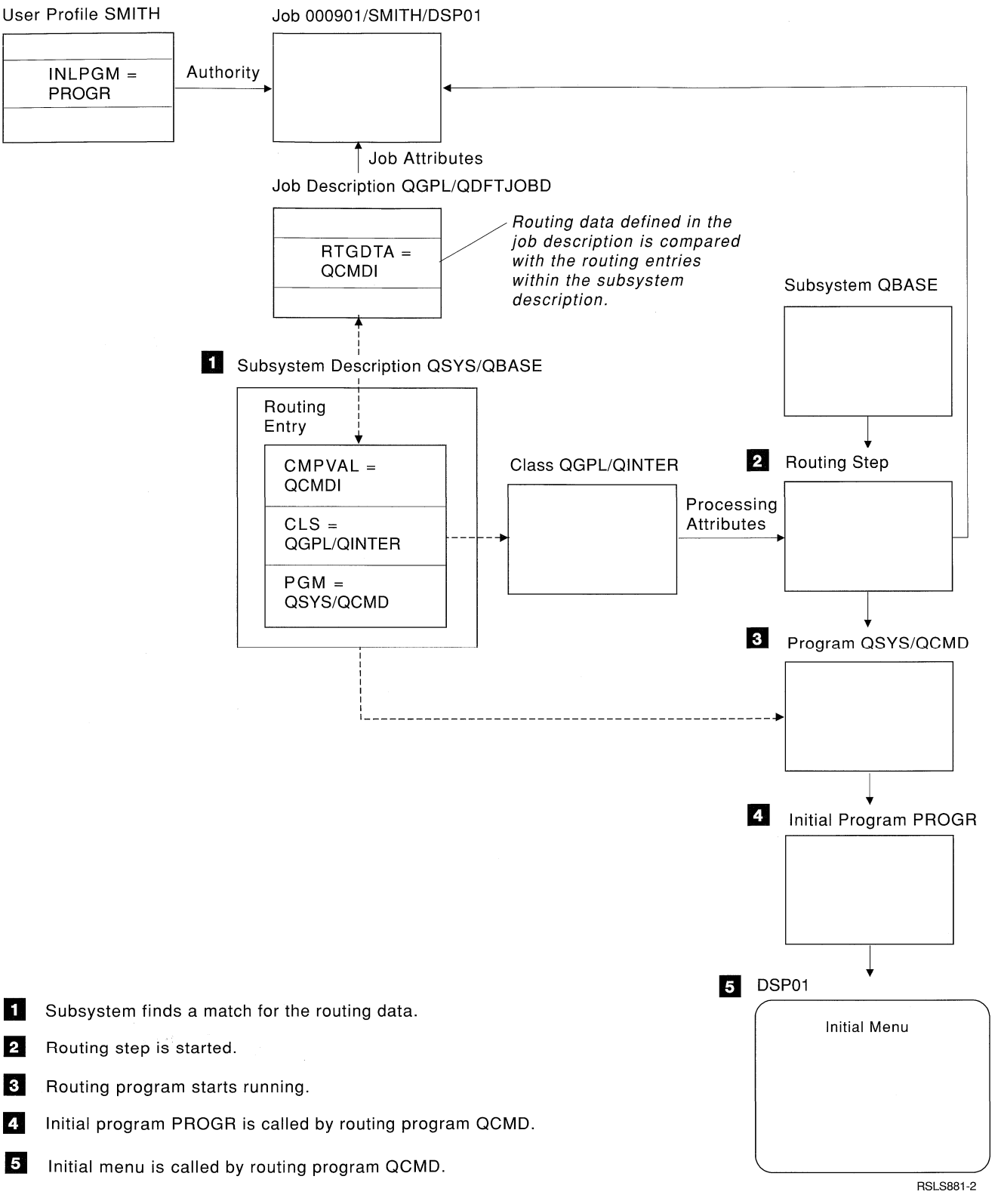


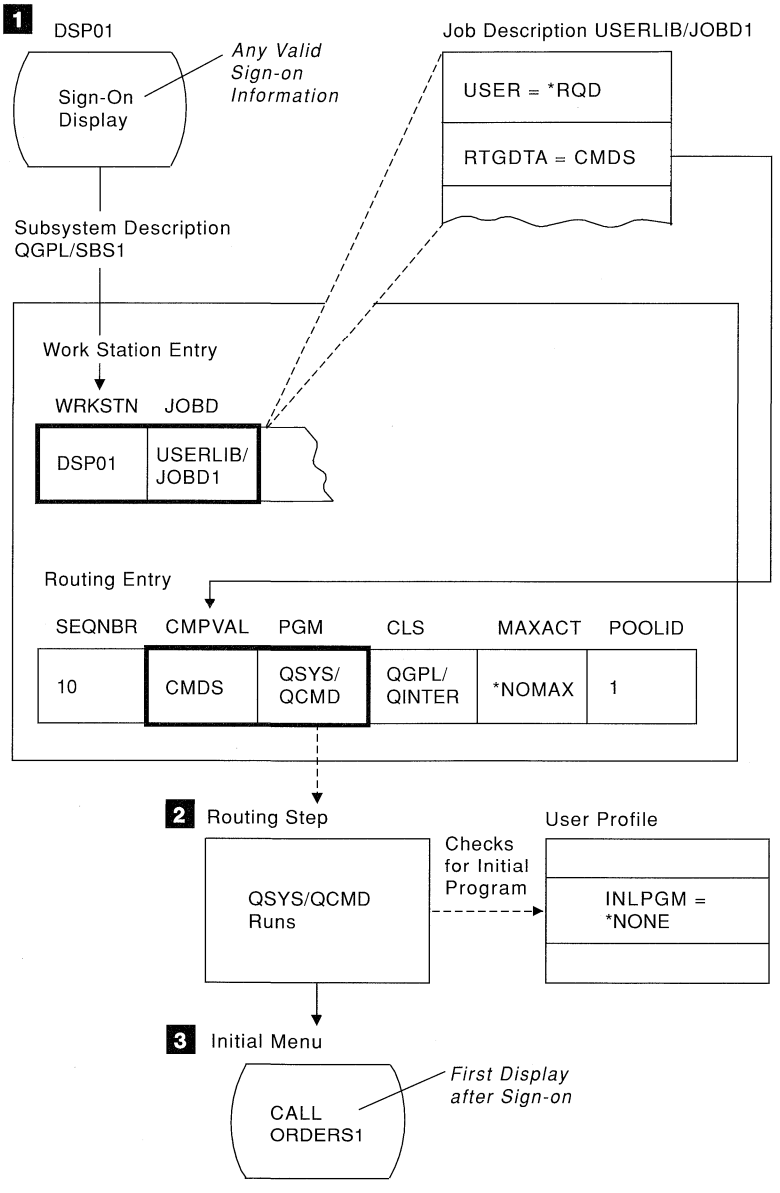
Figure 5-2. Subsystem Activity

The routing data is compared with the routing entries in the subsystem description. When a match is made, the program specified in the routing entry is called and the routing step is started. For the subsystem QBASE, the program QSYS/QCMD is called to process the commands in the job. QSYS/QCMD supports both interactive and batch jobs.

Interactive Job Illustrations

The following figures show techniques for building an environment in which a sample program USERLIB/ORDERS1 is called. (USERLIB/ORDERS1 could be an order entry program, an inquiry program, or a menu program.)

Figure 5-3 illustrates using the IBM-supplied program QSYS/QCMD to control the routing step.



1 A user signs on at work station DSP01.

The work station entry for DSP01 contains the name of the job description from which routing information is to be obtained.

The routing data CMDS matches the comparison value in the routing entry specifying program QSYS/QCMD.

2 QSYS/QCMD is called for the routing step and checks the user profile for an initial program.

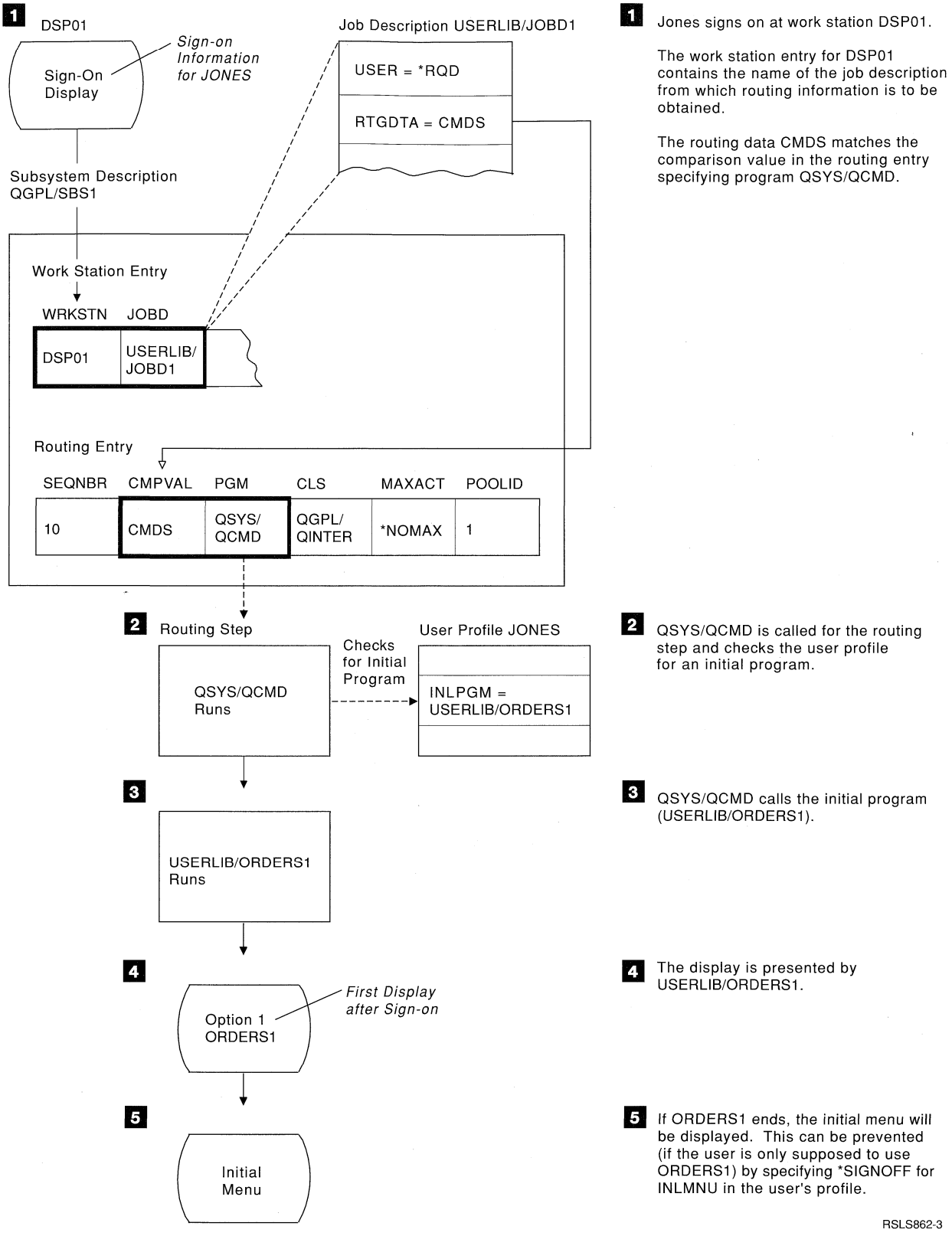
3 QSYS/QCMD displays the initial menu (no initial program specified).

The program USERLIB/ORDERS1 can be called from the main menu command line using the CALL command.

RSLS861-1

Figure 5-3. Interactive Jobs, Routing to QSYS/QCMD

Figure 5-4 on page 5-5 shows using a user program to control the routing step.



1 Jones signs on at work station DSP01.

The work station entry for DSP01 contains the name of the job description from which routing information is to be obtained.

The routing data CMDS matches the comparison value in the routing entry specifying program QSYS/QCMD.

2 QSYS/QCMD is called for the routing step and checks the user profile for an initial program.

3 QSYS/QCMD calls the initial program (USERLIB/ORDERS1).

4 The display is presented by USERLIB/ORDERS1.

5 If ORDERS1 ends, the initial menu will be displayed. This can be prevented (if the user is only supposed to use ORDERS1) by specifying *SIGNOFF for INLMNU in the user's profile.

RSLS862-3

Figure 5-4. Interactive Jobs, Calling User Program

Figure 5-5 on page 5-6 shows routing based on the user.

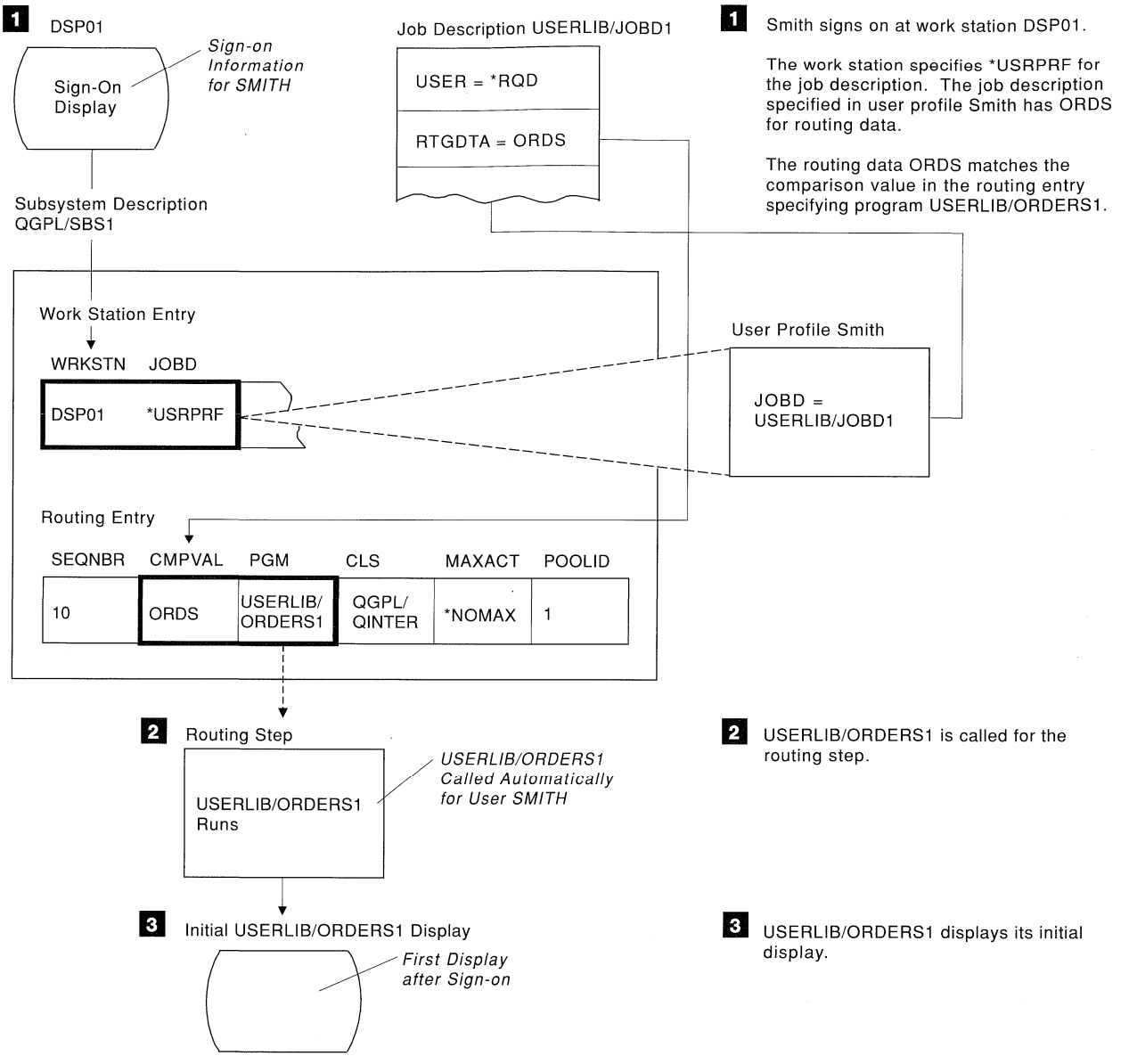


Figure 5-5. Interactive Jobs, Direct Routing to USERLIB/ORDERS1

Figure 5-6 on page 5-7 shows routing based on work station.

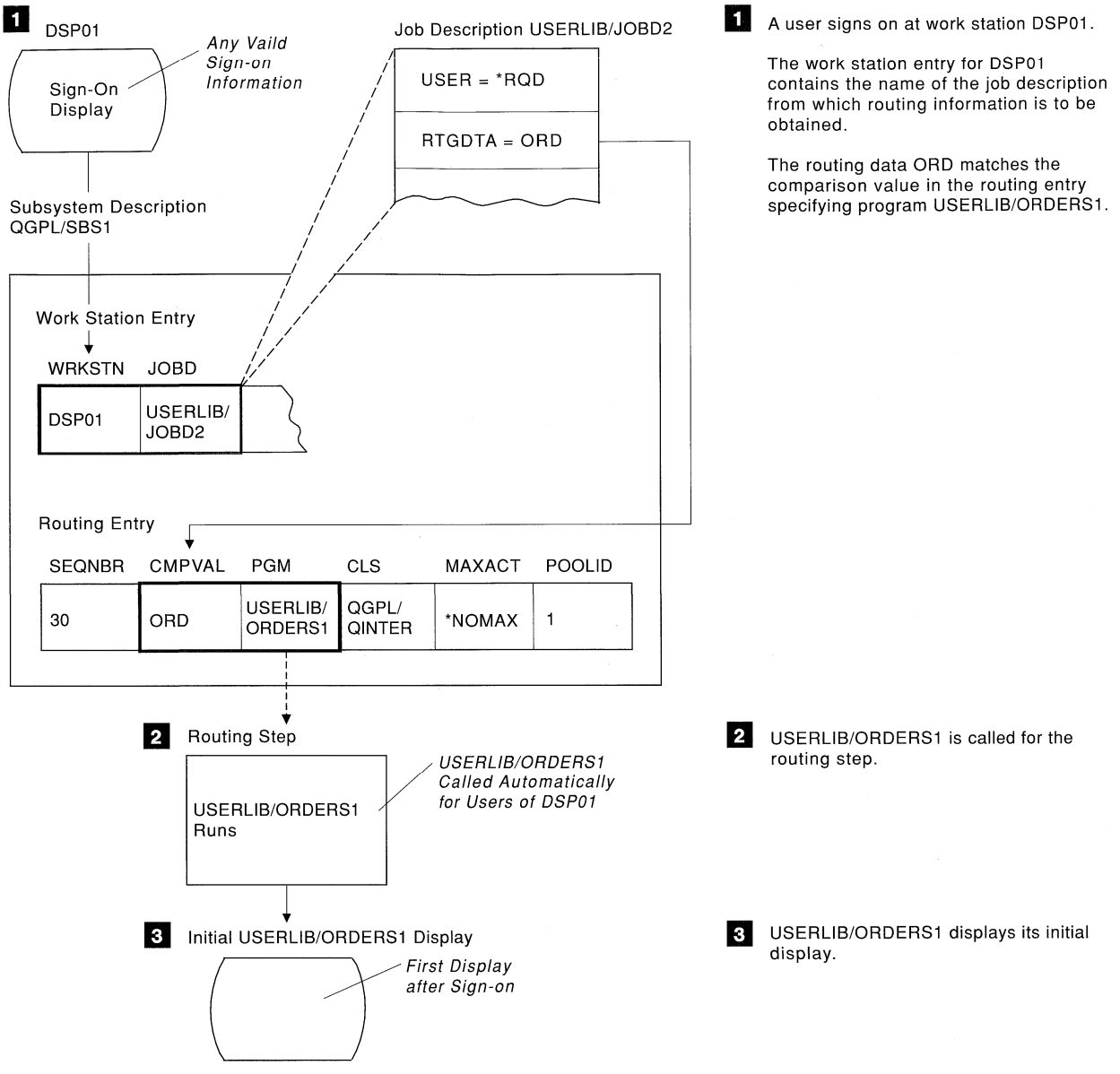


Figure 5-6. Interactive Jobs, Direct Routing to USERLIB/ORDERS1

Interactive Job Considerations

The preceding interactive job illustrations show the basic ways in which interactive work can be handled. In determining the best approach for a particular job, you must first determine which program should control the routing step:

- The IBM-supplied command processor QSYS/QCMD gives the greatest flexibility in terms of making functions available to work station users. Using QCMD gives you the following benefits:
  - The attention program is activated if specified in the user profile.
  - The initial program specified in the user profile is called.
  - The initial menu specified in the user profile is called.
  - The user is placed in System/36 environment if specified in the user profile.

  By default, using QCMD brings you to the Main Menu where you can enter commands directly, including the CALL command to call user-written functions. Menu options with online help are provided to give easy access to system functions. Also provided are

command selection menus, quick access to index search, and, if desired, the command entry function (called by CALL QCMD). The command entry functions are intended primarily for programmers and operators who require the full range of functions available through the direct use of commands.

- Your programs can be directly called to control the routing steps for interactive jobs. These programs can be designed to give a more specialized access to functions needed by your work station users than the IBM-supplied programs give. In addition, because your programs are tailored for specific functions, they should typically require even less system resource to support their running than the IBM-supplied programs. You may want to provide functions such as an initial program and initial menu also.

After you have determined which program controls the routing step, you must determine if routing is to be based on the work station from which the job was started, or on the user (user profile) who signed on. As shown earlier in this chapter, under "Interactive Job Illustrations" on page 5-4, routing based on the work station is accomplished using the routing data specified in the job description associated with the work station entry or profile for the device. Routing based on a user can be done using the initial program specified in the user profile or the job description in the user profile mapping to a routing entry other than QCMD.

Initial programs may interact with work stations to get input values from a work station user. When an initial program is called, it cannot receive parameter values.

An initial program can be used in one of two ways:

- It can be used to establish an initial environment for the user entering commands. For example, the library list can be changed or print files and message files can be overridden. When an initial program completes its function and returns to QSYS/QCMD, the initial menu is displayed.
- It can be used as the controlling program for the job. If the initial program does not return to QSYS/QCMD, it becomes the controlling program for the routing step. The initial menu is not displayed. The user can only request those functions available through the initial program.

  For example, a menu could be displayed with specific application options. The end user could only perform the functions on the menu. One of the options would normally be *sign off*. If the SIGNOFF command is run, the job ends and the system Main Menu is never displayed. If you use this approach, consider using the user profile option INLMNU to ensure no menu is displayed.

An initial program can be written so that when a return is issued, it either does or does not return to QSYS/QCMD. If the initial program returns to QSYS/QCMD, the initial menu is displayed. To avoid having QSYS/QCMD display the initial menu, the value *SIGNOFF can be specified for the initial menu in the user profile.

How To's

This section tells you how to perform some common tasks involving interactive jobs.

Ending Interactive Jobs

You can end interactive jobs two ways. To immediately end an interactive job, you can sign off the work station. Another way to end is by using the Disconnect Job Command.

Signing Off: The Sign Off (SIGNOFF) command allows you to immediately end an interactive job. To end the connection through the network, use the end connection parameter (ENDCNN) on the SIGNOFF command.

Disconnecting: The Disconnect Job (DSCJOB) command allows you to disconnect all jobs from a device. When the DSCJOB command is called, the job is disconnected and the sign-on display is shown again. To connect with the job again, sign on to the same device from which you disconnected. Another interactive job may be started on the device under a different user name.

An option on the System Request menu allows you to disconnect an interactive job, causing the sign-on display to appear. The option calls the DSCJOB command.

When connecting with a job again, the values specified on the sign-on display for program, menu, and current library are ignored.

A job which has PC organizer or PC text assist function active cannot be disconnected.

A TELNET job cannot be disconnected unless the user has used the system request to return to the client system from the server system.

All jobs will be disconnected for group jobs. When they are connected again, you return to the place where the disconnect was issued. If the last active group job ends before you connect again, you return to the next group job.

If the job cannot be disconnected for any reason, the job will be ended instead.

All disconnected jobs in the subsystem end when the subsystem ends. If a subsystem is ending, the DSCJOB command cannot be issued in any of the jobs in the subsystem.

The Disconnect Job Interval (QDSCJOBITV) system value can be used to indicate a time interval for which a job can be disconnected. If the time interval is reached, the disconnected job ends.

Disconnected jobs that have not exceeded the QDSCJOBITV value end when the subsystem is ended or when an IPL occurs.

The Device Recovery Action (DEVRCYACN) job attribute specifies what action to take when an I/O error occurs for a job's requester device. The DEVRCYACN attribute has the following options:

***SYSVAL**
Default, points to the system value QDEVRCYACN. The system value will support all of the values that the job attribute supports (except *SYSVAL).

***MSG**
Default for the system value. Signals the I/O error message and lets the application program perform an error recovery.

***DSCMSG**
Disconnect the job. Upon connecting again, a new error message signals the user's application program indicating the device was lost and recovered since the I/O, and the contents of the display need to be shown again.

***DSCENDRQS**
Disconnect the job. Upon connecting again, an end request function is performed to return control of the job to the last request level (similar to System/36 returning to the command processor).

***ENDJOB**
End the job. A job log is produced for the job. A message is sent to the job log and to the QHST log indicating the job ended because of the device error.

***ENDJOBNOLIST**
End the job. No job log is produced. A message is sent to the QHST log indicating the job ended because of the device error.

Sometimes, jobs end at the same time (for example, a remote line drops and the job attributes for the jobs attached to that line are set to *ENDJOB or *ENDJOBNOLIST). In addition to the job ending, the following actions occur:

- The job's priority is lowered. This occurs so the job is no longer at the same priority as other locally attached interactive jobs.
- The job's time slice is set to 100 milliseconds. This occurs to give higher priority jobs a better chance of getting processing resources.
- The job's purge attribute is set to *YES to free up as much main storage as possible for other jobs.

Automatically Ending Inactive Interactive Jobs

You can request the subsystem to send a message to a message queue when an interactive job has been inactive for a specified period of time. You, or a program monitoring that message queue, can then end or disconnect the job if desired. This control over inactive jobs provides security so that users do not leave signed-on display stations inactive.

The ADLINTJOBS parameter on the ENDJOB command allows you to end all of the interactive jobs associated with the work station, or all jobs associated with the group (if the job is a group job).

You can control the amount of time the work station can remain inactive before the subsystem sends a message (called time-out) by specifying a

time interval in the QINACTITV system value. At the end of the specified time interval, the subsystem checks the interactive jobs running in it to determine whether any of the work stations have been inactive for the entire interval. See "QINACTITV" on page 2-37 for a complete description of the system value QINACTITV.

A subsystem determines that a work station is inactive if all of the following are true:

- The job has not processed any additional transactions during the time interval.
- The job status is display wait.
- The job is not disconnected.
- The job status has not changed.
- The subsystem in which the job is running is not in the restricted state.

A transaction is defined as any operator interaction, like scrolling, pressing enter, pressing function keys, and so on. Typing at the work station without pressing enter is not considered a transaction. If a job at the work station, either secondary and/or group job, does not meet the inactive criteria, the job is considered active.

When an inactive job is found, the system value QINACTMSGQ is used to determine the processing options. The user can choose from the following processing options:

- Set the system value QINACTMSGQ to a message queue name. If you specify a message queue name for QINACTMSGQ, a user or program can monitor the message queue and take whatever action is necessary, such as ending the job.
- Set the system value QINACTMSGQ to *DSCJOB. If you specify *DSCJOB for QINACTMSGQ, the system disconnects all jobs at the work station (both group and secondary jobs). A message indicating all jobs at the work station have been disconnected is sent to QSYSOPR.

  If a work station with a secondary job pair is inactive, two messages (one for each of the secondary job pairs) are sent to the message queue. The user or program can then use either the ENDJOB command against one or both of the secondary jobs, or the DSCJOB command against the active job at the display.

  If an inactive job has one or more group jobs, a single message is sent to each of the message queues. (The message queue indicates whether or not the job is a group job.)

  A message continues to be sent for each interval that the job is inactive.
- Set the system value QINACTMSGQ to *ENDJOB. If you specify *ENDJOB for QINACTMSGQ, the system ends all of the jobs at the work station (both group and secondary jobs). A message indicating all jobs at the work station have ended will be sent to QSYSOPR.
- Set the system value QINACTMSGQ to *DSCJOB. If you specify *DSCJOB for QINACTMSGQ, the system disconnects all of the jobs at the work station (both group and secondary jobs). A message indicating all jobs at the work station have been disconnected will be sent to QSYSOPR.

Note: Source pass-through jobs, client VTM (virtual terminal manager) jobs, and 3270 device emulation jobs are excluded from the time-out because they always appear inactive. System/36 environment MRT jobs are also excluded since they appear as batch jobs.

Avoiding a Long-Running Function from a Work Station

To avoid a long-running function (such as save/restore) from a work station without tying it up, the system operator can submit the job to a job queue. The controlling subsystem description QSYS/QBATCH or QSYS/QBASE, which is supplied by IBM, has a job queue QSYS/QBATCH that can be used for this purpose. If you created your own controlling subsystem, you should refer to the job queue for that subsystem. The system operator can submit the commands from the system operator menu. The following is an example of submitting a long-running command:

```
SBMJOB JOB(SAVELIBX) JOBD(QBATCH) JOBQ(QSYS/QBATCH)
   CMD(SAVLIB LIBX DEV(DKT01)
```

Chapter 6. Group Jobs

This chapter describes how to start many interactive jobs at one work station using group job support. How to start, handle, and end group jobs is described along with a sample application for group jobs. This chapter also describes Attention key handling programs which allow you to go quickly from one group job to another.

Concepts

Group job support allows a user to start up to 32 interactive jobs at one work station. At any one time, only one group job can be active; the others are suspended. The group jobs are similar to secondary interactive jobs requested by pressing the System Request key; however, up to 16 group jobs can be started for each sign on at a work station (32 total when there is a secondary interactive job) and the application program can handle interruptions more flexibly. Figure 6-1 shows how group jobs can be started from a work station.

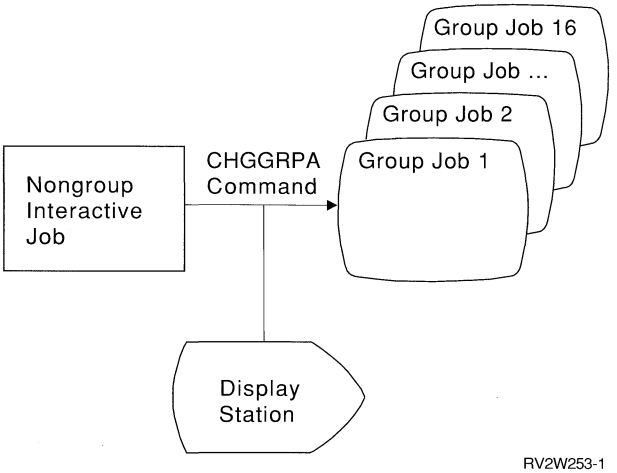


Figure 6-1. Starting Group Jobs from a Work Station

Attention key handling support makes it easy for the user to transfer control from one group job to another quickly, without ending one job to go to the other. To transfer control from one group job to another, the user presses the Attention key, and an Attention key handling program can either present a menu (from which the user chooses a group job) or immediately transfer the user to another group job.

The major advantages of group jobs are the following:

- The work station user can press the Attention key to interrupt work in one interactive group job, change to any of several other interactive group jobs, and return to the original group job quickly.

  The Attention key is made valid by the Set Attention Program (SETATNPGM) command and can be used independently of group jobs.
- Group jobs can give a significant performance advantage over alternative methods. If an Attention key handling menu is used to handle normal interruptions, the environment for processing these interruptions can be built on first use and then suspended. When a request is repeated, the function can be called with all files open and the proper display shown. This allows the process access group (PAG) for each function to be a minimum size, and the work station user does not have to exit a function to get to a menu from which to select another function.
- Using group jobs with display station pass-through provides a convenient and fast way to change among many interactive jobs on many different systems in a network. See the *Remote Work Station Guide* for more information on display station pass-through.

The important concepts to understand about group jobs are:

- Group jobs apply only to interactive jobs.
- Up to 16 group jobs can exist in one group (16 more are available if the user transfers to a secondary interactive job).
- Group jobs are unique to a user (they are not shared by multiple users).
- Only one group job at a time is active (the others are suspended).
- Each group job is independent and has its own job log, spooled files, library QTEMP, and so forth.
- A group job is called by the Transfer to Group Job (TFRGRPJOB) command. This command is typically run from a user-written menu

program, which is called by pressing the Attention key (the SETATNPGM command must have been previously run).

- A 512-byte group data area can be used to pass data between one group job and another. This group data area is implicitly created by the Change Group Attributes (CHGGRPA) command. The *CL Programmer's Guide* contains more information on group data areas.

Relationship of Group Jobs to a Secondary Interactive Job

Figure 6-2 shows the relationship of group jobs to a secondary interactive job.

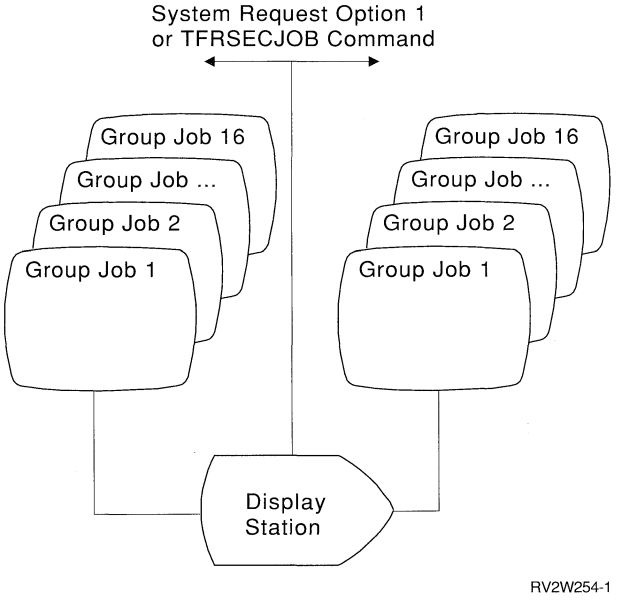


Figure 6-2. Group Jobs and a Secondary Interactive Job

In this case, the CHGGRPA command is run twice, once for each interactive job.

The Group Job function is similar to the System Request function in that there is only one job active at a time while the others are suspended. Group jobs differ from system request in the following:

- Starting a group job does not require signing on. The same user profile and environment are used.
- Up to 16 group jobs can exist at any one time. The user must select which group job to transfer to, whereas using system request permits the user to transfer between only two jobs. Normally in group jobs, a menu reached by pressing the Attention key allows the user to select which group job to transfer to. It is possible to use group jobs together with system request for a total of 32 group jobs available for a single user. However, these 32 jobs are in two separate groups, each group having its own group data area and other group attributes.
- The System Request function allows the work station user to suspend a job while the keyboard is locked and application functions are in progress. This can interrupt a logical sequence of events. For example, records may be left locked. In contrast, the Attention key is active only when the keyboard is unlocked for input. Also, the application can control when the Attention key is active, and prevent its use at inappropriate times. The System Request function is always available if the work station user has authority to it.

Sample Application for Group Jobs

To understand the group job concept, assume a work station user is shown a Main Menu with the following options:

```
     MAIN MENU

1. Post cash
2. Inquire into customer master file
3. Inquire into accounts receivable file


Option: _


F12=Cancel
```

The work station user selects option 1 (Post cash) to start entering payments into the file. While the work station user is entering data, he receives a telephone call requesting information that can be answered using option 2 (Inquire into customer master file).

Such an interruption can be handled in several ways:

- Have the work station user end the Post cash function, return to the Main Menu, and select the Inquiry function. When done with the inquiry, the work station user returns to the Main Menu, selects the Post cash function and continues. This requires system interactions each time the work station user selects a different function and may require additional system resources for opening and closing files.
- Code the inquiry function as part of the Post cash function. This method is usually done by specifying that a command function key be enabled to request the inquiry. This can be awkward if the same Inquiry function must be accessed from multiple programs.
- Press the System Request key and start a secondary interactive job. This can be a good solution if the work station user uses only one function during an interruption. However, if several functions are used, this solution can become awkward.
- Use the group job approach as shown in Figure 6-3 on page 6-4.

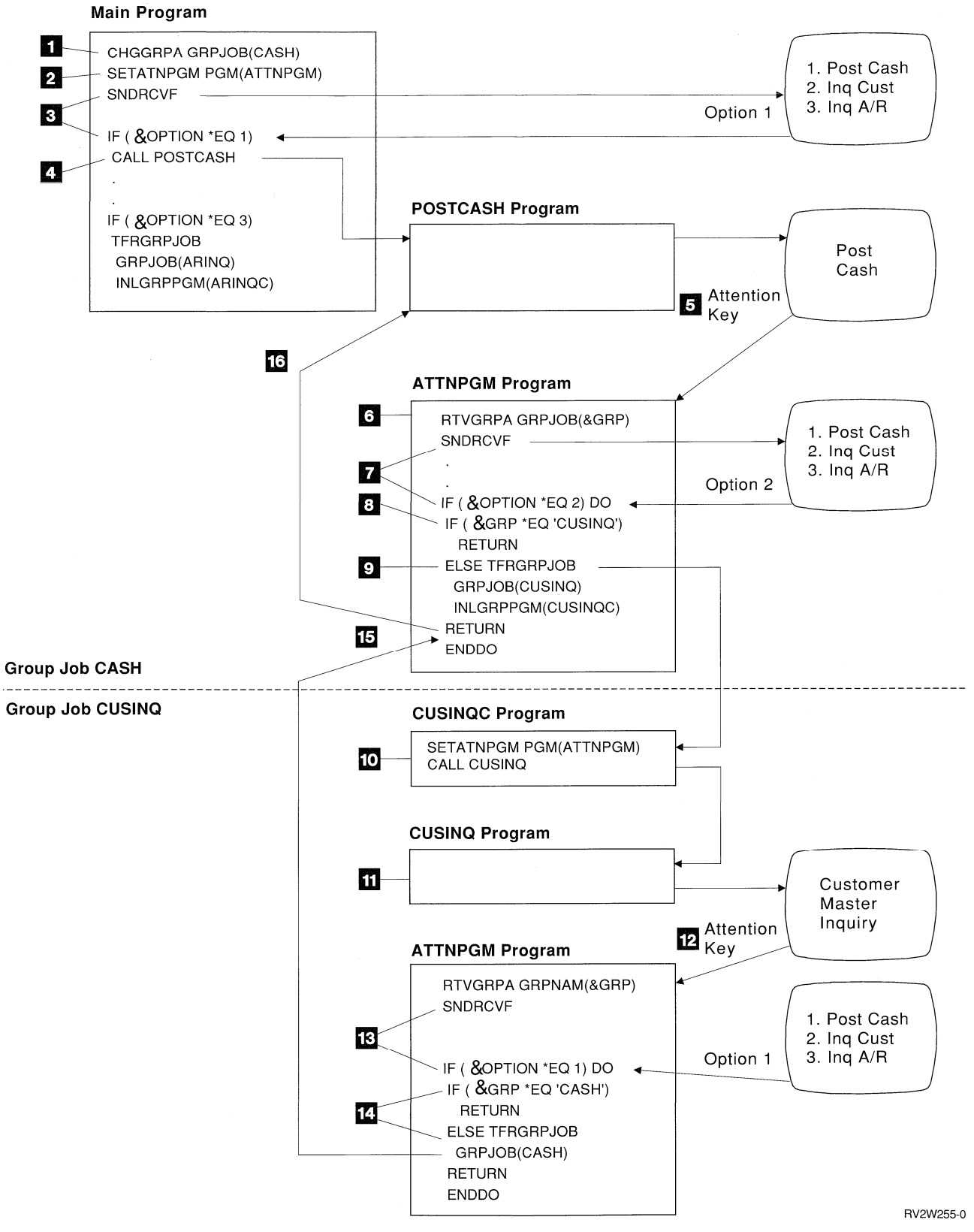


Figure 6-3. Group Job Approach

The following steps illustrate how the sample application works:

1 The Change Group Attributes (CHGGRPA) command changes the current nongroup interactive job to a group job named CASH.

2 The Set Attention Program (SETATNPGM) command specifies that the system should recognize the Attention key and call the ATTNPGM program when the Attention key is pressed.

3 The SNDRCVF command displays a menu from which the work station user selects functions to work on. In this case, the work station user selects option 1 (Post cash).

4 The main program calls the POSTCASH program, and the work station user starts posting cash transactions.

Note: The main program runs a CALL command to the post cash program and a TFRGRPJOB command to the other functions. This allows the main job to always be known as CASH, but other techniques can be used. If the work station user chooses to *end* one of the group jobs other than CASH (instead of pressing the Attention key to suspend it), the system automatically returns to the previously active group job.

5 To change from posting cash, the work station user presses the Attention key. The system saves the current contents of the display and calls the Attention key handling program (ATTNPGM).

6 The Attention key handling program (ATTNPGM) is written to be used by any of the jobs in this group. The program retrieves the current group job name using the Retrieve Group Attributes (RTVGRPA) command.

7 The Attention key handling program displays a menu from which the work station user can select a function. The menu can contain any options, but in this example, the options are the same as those shown on the Main Menu. The work station user selects option 2 (Inquire into customer master file).

8 If the option is for the current group job (for option 2 this is CUSINQ), the program returns to the current group job. For example, assume the operator posting cash pressed the Attention key, then decided not to change group jobs and continue posting cash. Instead of transferring to a group job, only the RETURN command is needed.

9 If the option is for a different group job, the program runs a TFRGRPJOB command to suspend the current job and activate a new group job and program. For option 2, the TFRGRPJOB command specifies the name of the group job to be started (in this example, CUSINQ), and which program to call (CUSINQC program).

10 The group job CUSINQ is activated and the initial group program (CUSINQC) is called. The job has already been identified as a group job by the TFRGRPJOB command; therefore the CHGGRPA command is not needed. This group job also uses the SETATNPGM command to allow for an interruption. The same program (ATTNPGM) is used, but it runs in a different group job.

11 The work station user inquires into the customer master file and is now ready to return to the post cash function.

12 The work station user again presses the Attention key. The system saves the current contents of the display and calls the Attention key handling program (ATTNPGM). The user sees the same Attention Handling menu, but is in a different group job.

13 The work station user selects option 1 (Post cash).

14 Because the name of the current group job is CUSINQ (not CASH), the TFRGRPJOB command is run to transfer to the CASH group job.

15 In this instance, the group job is already active but suspended; therefore the system resumes the suspended group job. The ATTNPGM program is continued at the next instruction following the TFRGRPJOB command that was run in step 9. This causes a return to the POSTCASH program.

16 The previous display (which the system saved when the work station user pressed the Attention key in step 5) is restored, and the work station user continues posting cash transactions where he left off.

If the work station user wants to inquire into the customer master file again, the user would again press the Attention key and receive the attention

handling menu. Because the group job would already be active, the system would resume the job where it was suspended (that is, at the instruction following the TFRGRPJOB command in step 14). This return causes a return to the CUSINQ program where it was interrupted when the work station user pressed the Attention key in step 12.

If the work station user wants to inquire into the accounts receivable file, the same set of steps occurs. However, the work station user does not have to return to the post cash function first. The attention handling menu allows him to select any of the functions in any sequence. Note that the work station user cannot start two group jobs with the same group job name. If the job specified a group job name that is already active, control is passed to that job (no new group job is started) and the initial group program parameter is ignored. It is also possible to have several group jobs (having unique names) all doing the same function.

The Attention key should not be pressed until the program requests input from the work station. This allows the program to prevent an interruption during certain functions; for example, when records are locked. See "Designing an Attention Key Handling Program" on page 6-10 later in this chapter.

When the Attention key is pressed, the system saves the current display. The current display is not saved when a TFRGRPJOB command is run.

The complete printout for this example is shown later in this chapter.

Effect of Call Level on Attention Key Status

You can use the Set Attention Program (SETATNPGM) command with SET(*ON) specified to identify a program as the Attention key handling program at that call level in the job running the command. When the Attention key is pressed, the running job is interrupted, the display is saved, and the Attention key handling program is called. No parameters are passed to the Attention key handling program when it is called.

The Attention key handling program runs in the same job and has the same job attributes, overrides, and group authorities as the program that issued the SETATNPGM command. However, program-adopted authority does not originate from the program that was interrupted.

The SETATNPGM command is call-oriented. That is, a SETATNPGM command issued at one call level causes the Attention key handling program to be in effect at the current call level as well as lower call levels, until another SETATNPGM command is run to change the Attention key handling program or Attention key status. Whenever a program that issued a SETATNPGM command returns, the display is restored and the Attention key handling program and Attention key status are reset to what they were before the current call. If a Transfer Control (TRFCTL) command is used instead of a RETURN command, the status is not reset until the program that was transferred to returns.

You may also specify an Attention key handling program in the user profile. The *Security Reference* manual contains information on this.

Figure 6-4 on page 6-7 illustrates the use of the SETATNPGM command at different call levels.

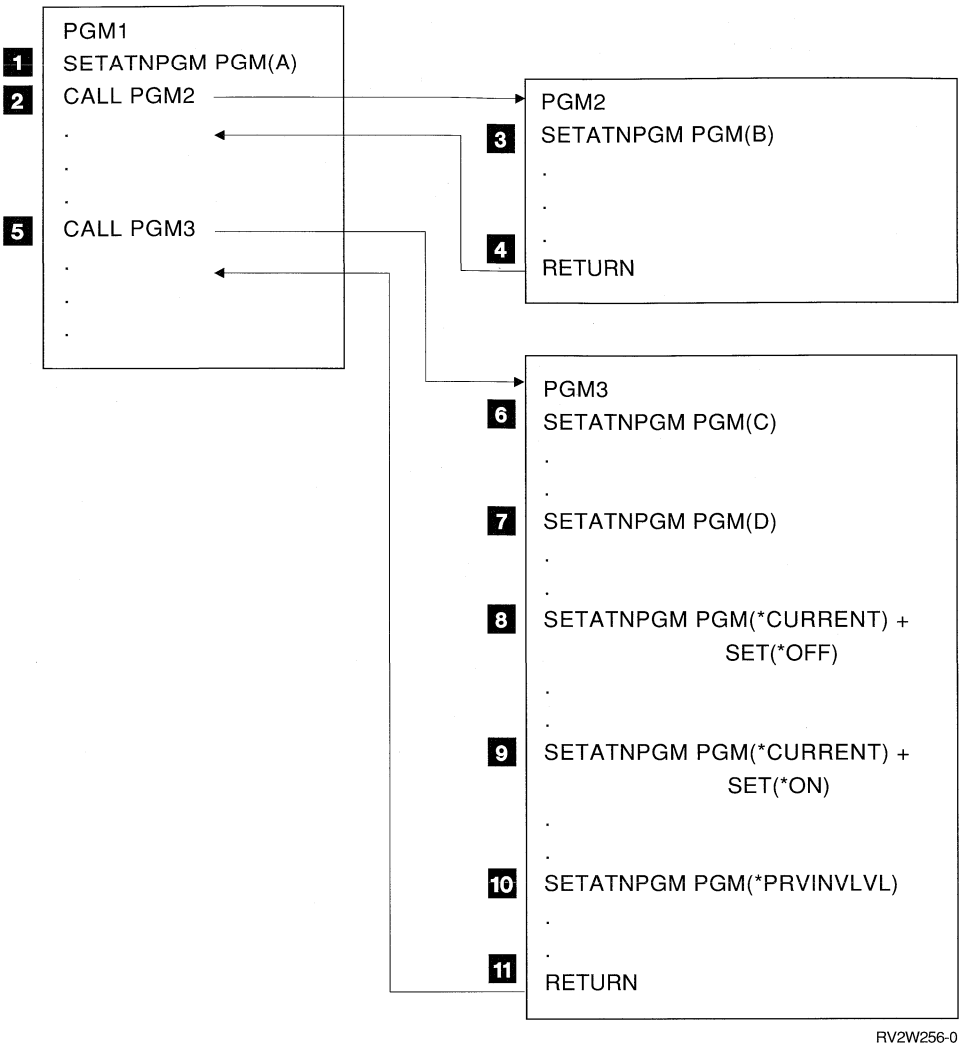


Figure 6-4. SETATNPGM at Different Call Levels

1 PGM1 issues a SETATNPGM command which causes program A to become the Attention key handling program.

2 PGM1 then calls PGM2. Program A continues to be the Attention key handling program until step 3.

3 Another SETATNPGM command causes program B to become the current Attention key handling program.

4 After returning from PGM2, program A again becomes the Attention key handling program.

5 PGM1 calls PGM3.

6 The first SETATNPGM command issued in PGM3 changes the Attention key handling program from program A to program C.

7 The second SETATNPGM command issued in PGM3 changes the Attention key handling program to program D.

8 Specifying SETATNPGM PGM(*CURRENT) SET(*OFF) causes no program to be called when the Attention key is pressed. This allows the program to complete a series of interactions with the work station user without being interrupted by the Attention key.

9 Specifying SETATNPGM PGM(*CURRENT) SET(*ON) causes program D to be called when the Attention key is pressed.

10 The Attention key handling program that was in effect at the previous recursion level goes into effect. That is, program A becomes the Attention key handling program again and the Attention key is set to *ON.

11 When PGM3 returns, program A continues to be the Attention key handling program.

Conditions for Using the Attention Key

In normal work station use, the Attention key can be pressed only when the keyboard is unlocked; that is, the program is ready for input. This occurs when a read or write-read operation is issued or the UNLOCK DDS keyword is used in a write operation. The use of the Attention key differs from that of the System Request key in that the application program has control over when it can be interrupted.

An exception to this occurs with application programs performing a get-no-wait operation on multiple device files. Pressing the Attention key causes these programs to be interrupted at any point by the Attention key handling program. (Even though the input inhibited light may be on, the keyboard is unlocked during a get-no-wait operation.) Application programs performing sensitive functions (especially during a get-no-wait operation) should therefore be protected by running SETATNPGM PGM(*CURRENT) SET(*OFF) before and SETATNPGM PGM(*CURRENT) SET(*ON) after sensitive code.

Note: A high-level language program can use the SETATNPGM command by calling QCMDEXC.
The Attention key cannot be used to call an Attention key handling program when the following conditions exist:

- The keyboard is locked. (Note the exception described earlier for get-no-wait operations.)
- The System Request menu or any of its options is being used.
- The display message display is shown.
- The OS/400 licensed program is already calling the Attention key handling program which makes it already active; however, if the program issues another SETATNPGM, the Attention key is enabled.
- A BASIC session is in progress, or a BASIC program is called.

Note: In a BASIC session, the Attention key is handled by BASIC, as appropriate. For example, if a BASIC program is called after a SETATNPGM command has set the Attention key on, the Attention key is handled by BASIC. After the BASIC program ends, your Attention key handling program takes effect again.

Guidelines for Coding Attention Key Handling Programs

Caution is necessary when defining an Attention key handling program because the Attention key handling program runs in the same job as the program that is in progress when the Attention key is pressed. Therefore, the interrupted program is not protected by any locks it held. If the interrupted program has an exclusive lock on an object, the Attention key program, because it runs in the same job, is part of the job that has the exclusive lock.

The following guidelines are recommended for defining Attention key handling programs:

- Use simple functions such as menus that allow the work station user to transfer to another group job or to a secondary interactive job.
- Avoid referring to objects or functions that may be in use when the Attention key is pressed.
- Avoid calling nonrecursive functions when the Attention key is pressed. Nonrecursive functions are functions that cannot be interrupted, then called again. Many functions, such as high-level language programs and utilities like DFU, are nonrecursive.
- Avoid giving an option that allows the work station user to display the command entry display as part of the current job. For users who are programmers, it is meaningful to display a menu that includes an option for the command entry display. The command entry display should be specified as a separate group job (for example, by specifying INLGRPPGM(QCMD) on the TFRGRPJOB command). This avoids re-using objects already in use.
- Attention key handling programs do not have the authority adopted by the program that was in progress before the Attention key was pressed.
- Attention key handling programs do not have their own data area (*LDA). Since there is

only one local data area per job, and the Attention key handling program runs in the same job as the interrupted program, both programs share the same local data area.

- Be aware that a read-from-invited devices operation could time-out during the time that the Attention key handling program is running. Therefore, if a time-out were to complete in the program in progress while the Attention key handling program is running, whatever action taken as a result of that time-out will occur on return to the program in progress.

  For example, if the following conditions are met the program will exit on return from the Attention key handler:

  - The WAITRCD value in the file is set to 60 seconds.
  - The program is set to exit if a key is not pressed in one minute.
  - The Attention key program is called and runs longer than that minute.

  However, caution should be used, since a check for available data is made before checking that the time-out has completed. If a key is pressed immediately after leaving the Attention key handler, data could be available that could complete the read-from-invited devices and the time-out would not be checked. This could cause unexpected results.

Performance Considerations

Each group job (active or suspended) requires approximately 1KB of dedicated main storage in the machine pool. Thus, the number of group jobs that are allowed to be active is a consideration. However, if main storage is not limited, this may be a good way to gain the benefits of separate process access groups (PAGs). This also allows you to avoid the overhead caused by repetitive opening of files and reestablishing environments, and to avoid excessive interactions to access common functions. For information on the machine pool, see Chapter 13, "Performance Tuning."

The effect on the system for a large number of suspended jobs is normally small if the dedicated main storage requirement is not a factor.

When a TFRGRPJOB command runs and a new job must be started, the overhead involved is roughly the same as signing on to the system. When the command is run and the group job is already started, the overhead required is roughly the same as using the transfer to a secondary job option on the System Request menu when the secondary job is already active.

If a group job is to be run with any frequency, it is desirable to prevent it from ending. That is, do not end the program, but issue a TFRGRPJOB command to prevent job starting each time the group job function is needed.

The SETATNPGM command causes the current display to be saved when the Attention key is pressed, and to be restored when the Attention key handling program ends. This is roughly the same as using of the System Request menu and has a more noticeable effect on remote work stations.

The controls on the number of jobs active in the system (the MAXJOBS parameter on the CRTSBSD command) are not affected by the number of group jobs active at any time. However, all system values that control the creation of job structures (QACTJOB and QADLACTJ, and QTOTJOB and QADLTOTJ) are affected; these values may need to be increased to allow for the addition of group jobs.

How To's

This section tells you how to perform some common tasks involving group jobs.

Starting, Handling, and Ending Group Jobs

When working with group jobs, you can use the following commands:

- The CHGGRPA command changes your nongroup job to a group job and switches a group job back to a nongroup job (if it is the only job in the group).
- The TFRGRPJOB command switches from one group job to another group job in the same group and creates new group jobs. After each use of the TFRGRPJOB command,

the SETATNPGM command must be used to set the Attention key on, if desired.

- The ENDGRPJOB command ends one group job in a group.
- The CHKRCDLCK command checks if the job has any record locks before transferring to another group job while you are in an update operation.
- The SIGNOFF command ends all group jobs in the group.
- The ENDJOB command supports the parameter ADLINTJOBS. If *GRPJOB is specified and the job specified on the JOB parameter is a group job, all jobs associated with the group end.

The CHGGRPA command identifies the current job as a group job and gives it a group job name to uniquely identify it in the group. (At this point the group has only one group job.) Each group job is unique for a user. Two different users do not share the same group job. When a job is designated as a group job, it then has the capability to call a new group job. There are also restrictions on group jobs (such as RRTJOB, TFRJOB may not be used). When there is only one active job in the group, that job can become a nongroup job.

To allow group jobs to communicate with each other, a special 512-byte data area called a group data area is automatically created when a job becomes a group job. The group data area can only be accessed by jobs in the group by using the special value *GDA in the DTAARA parameter of the data area command.

The use of group jobs does not require an Attention key menu approach as described in this section. A group job can be called from any application program or by the GRPJOB(*SELECT) parameter on the TFRGRPJOB command.

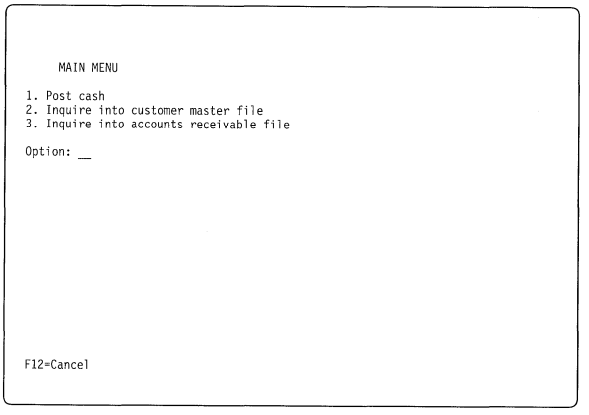
Designing an Attention Key Handling Program

There are essentially three approaches to designing an Attention key handling menu:

- Fixed menu. The Attention key handling program allows only a fixed set of options.
- Dynamic menu. The work station user determines the options.
- Combination approach. The work station user selects from a fixed set of options and can dynamically select additional group jobs.

Fixed Menu: This is the method used in the example earlier in this chapter. In that example, the user can only select from the options on the menu.

A menu is displayed to the work station user with the following options:



The program that displays the menu is coded as follows:

```
      PGM       /* First program */
      DCLF       MAIND
      CHGGRPA    GRPJOB(CASH) TEXT('Post cash')
      SETATNPGM  PGM(ATTNPGM)
LOOP: SNDRCVF
      IF    (&IN91 *EQ '1') RETURN /* F12 */
      IF    (&OPTION *EQ 1) CALL POSTCASH
      IF    (&OPTION *EQ 2)
              TFRGRPJOB GRPJOB(CUSINQ) +
              INLGRPPGM(CUSINQC) +
              TEXT('Customer inquiry')
      IF    (&OPTION *EQ 3) TFRGRPJOB
              GRPJOB(ACRINQ) + INLGRPPGM(ACRINQC) +
              TEXT('Accounts Receivable inquiry')
      GOTO       LOOP
      ENDPGM
```

In this example, the CHGGRPA command changes the current job into a group job. The SETATNPGM command specifies that the Attention key is allowed, and it specifies the Attention key handling program to be called (ATTNPGM). Because this program will display a menu, it is called an Attention key menu program. If the work

station user selects option 1 (Post cash), the program POSTCASH is called.

If another option is specified, a TFRGRPJOB command, which names the group job, is used. The system determines if the group job already exists. If it does not, the group job is started and the initial group program is used. If the group job already exists, the system transfers to that group job at the point where it was previously suspended. That is, it transfers to the next instruction following the TFRGRPJOB command that caused a transfer back to the CASH group job. The program loops back to the SNDRCVF command to redisplay the menu if control is ever transferred back to CASH.

When the work station user presses the Attention key, the Attention key menu program (ATTNPGM) is called. The display may differ from the initial menu, but in this example, the displays have the same options. Only the title differs to help the user identify the functions being used.

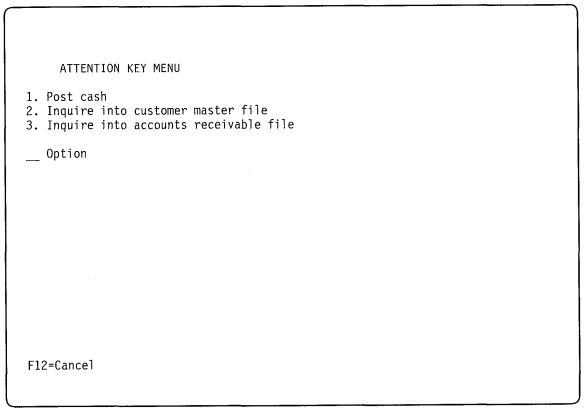


Figure 6-5 shows how the Attention key menu program (ATTNPGM) is coded.

```
PGM       /* Attention key menu program */
DCLF      ATTNMNUD
DCL       &GRP *CHAR LEN(10)
CHKRCDLCK
 MONMSG MSGID(CPF321F)  EXEC(DO)
   SNDPGMMSG     MSG('Attention not allowed.
                     Complete transaction')
 RETURN
ENDDO
RTVGRPA   GRPJOB(&GRP)          /* Retrieve
                                   group name */
SNDRCVF
IF        (&IN91 *EQ '1') RETURN /* F12 */
IF        (&OPTION *EQ 1) DO     /* Post cash */
IF        (&GRP *EQ 'CASH') RETURN
          ELSE TFRGRPJOB GRPJOB(CASH)
          ENDDO
IF        (&OPTION *EQ 2) DO /* Cust inquiry */
          IF (&GRP *EQ 'CUSINQ') RETURN
          ELSE TFRGRPJOB  GRPJOB(CUSINQ) +
                          INLGRPPGM(CUSINQC) +
                          TEXT('Customer inquiry')
          ENDDO
IF        (&OPTION *EQ 3) DO /* Acct rec inquiry */
          IF (&GRP *EQ 'ACRINQ') RETURN
          ELSE TFRGRPJOB GRPJOB(ACRINQ) +
               INLGRPPGM(ACRINQC) +
               TEXT('Accounts Receivable inquiry')
          ENDDO
ENDPGM
```

Figure 6-5. Attention Key Menu Program

The Attention key menu program may differ for any group job, but in this case the same program is used for all group jobs. Because the work station user may press the Attention key, then request the same function that was interrupted, the program retrieves the current group job name and compares it to the group job name for the option that was selected. If they are the same, a RETURN command is issued. This means that the work station user was in a function, pressed the Attention key, then decided to continue the function. If the names differ, the work station user is requesting a different group job and the TFRGRPJOB command is used. The system determines if the group job is already active; if it is not, the system starts the group job and calls the initial program. If the group job is already active, the system transfers to where it left off. Note that the CASH group job was originally made active and therefore the INLGRPPGM parameter is not needed.

When a group job is started, the system automatically establishes much of the environment by using the attributes from the transferring group job. Thus functions such as the library list and the logging level need not be set if the same values are desired. If however, the new group job also requires the use of the Attention key, it must be specified on a SETATNPGM command. For example, the initial program for the CUSINQ group job is CUSINQC as follows:

```
PGM         /* CUSINQC program */
SETATNPGM   PGM(ATTNPGM)
CALL        CUSINQ
ENDPGM
```

In the previous example, the same Attention key menu program is specified and then the processing program (CUSINQ) is called. If the work station user ends the CUSINQ program, the CUSINQC program also ends, and the system automatically ends the group job and transfers back to the group job that was previously active. It would be possible to issue a TFRGRPJOB GRPJOB(*PRV) after the CALL and then a GOTO to repeat the CALL command. This will keep the group job in a suspended state. This decreases the time necessary to transfer to this job on the next call of the CUSINQ option.

Dynamic Menu: In this case the Attention key menu program runs the TFRGRPJOB command with GRPJOB(*SELECT) specified. This provides a display of the current group jobs and allows the work station user to transfer to any group job. Thus the work station user can decide what functions are needed. This approach can be useful for programmers because it allows them to call several group jobs and to specify what command should be run.

For example, assume that the work station user specifies the following commands or runs them as part of a standard setup program:

```
CHGGRPA     GRPJOB(NORMAL) TEXT('Normal job')
SETATNPGM   PGM(MENU1)
```

Program MENU1 would be coded as:

```
PGM
TFRGRPJOB   GRPJOB(*SELECT)
ENDPGM
```

Assume the work station user is working in the CASH group job, with two other group jobs suspended. If the work station user presses the Attention key, the Transfer to Group Job display appears as follows:

```
                          Transfer to Group Job
                                                          System:   XXXXXXXX
Active group job . . . . :   CASH
Text . . . . . . . . . . :   Post cash

Type option, press Enter.
  1=Transfer group job


      -----------------Suspended Group Jobs-------------------------
Opt   Group Job     Text
 _    ACRINQ       Accounts receivable inquiry
 _    CUSINQ       Customer inquiry









                                                                  Bottom
F3=Exit   F5=Refresh   F6=Start a new group job   F12=Cancel
```

The work station user could then press the F6 key to enter a TFRGRPJOB command with prompting. When the TFRGRPJOB prompt is displayed, the work station user could specify a GRPJOB parameter value and either of the following for the INLGRPPGM parameter:

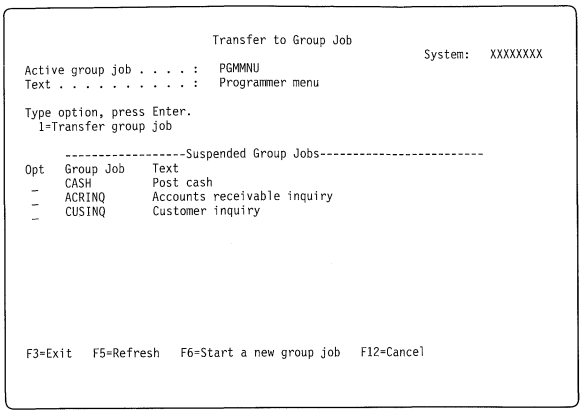
- QCMD, that would display the initial menu.
- A standard program that would build the environment the work station user wants and specify the same Attention key handling program. For example, assume the work station user specified the command:

```
TFRGRPJOB   GRPJOB(PGMMNU) INLGRPPGM(PGMMNUC)
            TEXT('Programmer Menu')
```

The PGMMNUC program could be coded as follows:

```
SETATNPGM   PGM(MENU1)
STRPGMMNU   .....
```

The work station user can now use the programmer menu. If the Attention key is pressed again, the display would show:



On this display, the jobs are shown in the order they were called.

Thus each group job that is already started can be easily called again, and the work station user can continue to add group jobs as required.

Combination Approach: This approach allows the following variations:

- The work station user can select from a fixed set of menu options. One option allows the work station user to prompt for the TFRGRPJOB command. The user could then specify the group job required or could specify GRPJOB(*SELECT) to receive a display of the currently active group jobs. If a user-written menu is displayed, an input field on the menu for the group job name could also be considered. A modification of this is for the Attention key menu program to extract the active group jobs using RTVGRPA command and dynamically build the menu choices (including the standard choices).
- The work station user can select a menu option to activate a standard set of group jobs. Then, when the user presses the Attention key, the Attention key program runs the TFRGRPJOB command with GRPJOB(*SELECT) specified. This would display all the standard programs and allow the user to add group jobs.

The following section illustrates the second variation.

An Approach for Programmers: You may want to set up several standard group jobs that are activated at sign-on. You can add additional group jobs if necessary. In Figure 6-6, you have the following standard group jobs.

Figure 6-6. Standard Group Job

Description	Group Job Name
Main group job	MAIN
Second programmer menu	PGMMNU2
Command entry	CMDENT1
SAA* OfficeVision/400* word processing function	STRWP1

In this approach, each programmer has a unique initial program. Figure 6-7 on page 6-14 shows an initial program for a programmer named SMITH.

This program does the following:

1. Establishes the environment (for example, library list).
2. Calls the standard group jobs (three group jobs plus the main group job are shown here). The group job data area is used to pass the command to be run. For each group job except the initial one, the same initial program (STDGRPC) is used.
3. The SETATNPGM command sets the Attention key on and establishes the ATTNPGM program as the standard Attention key handling program. When the Attention key is pressed, the TFRGRPJOB command with GRPJOB(*SELECT) specified displays the group job selection display. From this display, the programmer can change and add additional group jobs.

Note: To add another standard group job requires only two additional commands be added to the initial program and no changes to the other programs. With this approach, the group jobs would be started at sign-on and would not be ended unless the user signed off or entered the ENDGRPJOB command.

The initial program (STDGRPC) for the group jobs other than the main group job is shown in Figure 6-8 on page 6-14.

When the group job is started, it does the following:

```
   PGM
1
   CHGLIBL    LIBL(SMITH QGPL QPDA QTEMP QIDU)
   CHGGRPA    GRPJOB(MAIN) TEXT('Main group job')
2
   CHGDTAARA  DTAARA(*GDA) VALUE('STRPGMMNU
              SRCLIB(SMITH) + OBJLIB(SMITH)
              JOBD(SMITH)')
   TFRGRPJOB  GRPJOB(PGMMNU2) INLGRPPGM(STDGRPC) +
              TEXT('Programmer Menu # 2')
   CHGDTAARA  DTAARA(*GDA) VALUE('CALL QCMD')
   TFRGRPJOB  GRPJOB(CMDDENT1) INLGRPPGM(STDGRPC) +
              TEXT('Command entry # 1')
   CHGDTAARA  DTAARA(*GDA) VALUE('STRWP')
   TFRGRPJOB  GRPJOB(STRWP1) INLGRPPGM(STDGRPC) +
              TEXT('Word Processing # 1')
3
   SETATNPGM  PGM(ATTNPGM)
LOOP:
   STRPGMMNU  OBJLIB(SMITH) SRCLIB(SMITH)
              JOBD(SMITH)
   MONMSG     MSGID(CPF2320) /* F3 from
              Pgmrs Menu */
   GOTO       LOOP
   ENDPGM
```

Figure 6-7. Initial Program for SMITH

1 Retrieves the group data area, which contains the command to be run. The command is not run immediately, but is stored in a variable in the program. The program returns to the previous group job. The user does not see any displays, but would notice the overhead to build the group job. Assume at a later point the user selects the group job STRWP1 from the Group Jobs Selection menu. The user transfers to the STRWP1 group job. Because it is already active, it continues with the instruction following the TFRGRPJOB command.

2 The SETATNPGM command sets the Attention key on and establishes the ATTNPGM program as the Attention key handling program.

3 The program calls the QCMDEXC program and passes variable &CMD, which contains the STRWP command (see the initial program shown earlier). To exit the OfficeVision/400 word processing function, the programmer can do either of the following:

- Press the Attention key and receive the Group Jobs Selection menu. This allows the programmer to transfer to an existing group job or start a new group job.
- Press the Exit key from the OfficeVision/400 word processing function primary menu. This returns to program STDGRPC, which transfers back to the previous group job.

  Note: Pressing the Exit key does not end the STRWP1 group job. If the STRWP1 group job is again activated, the program returns to the next instruction, which loops back and runs the STRWP command again by calling QCMDEXC.

```
   PGM
   DCL        VAR(&CMD) TYPE(*CHAR) LEN(512)
1
   RTVDTAARA DTAARA(*GDA) RTNVAR(&CMD)
   TFRGRPJOB GRPJOB(*PRV)
2
   SETATNPGM PGM(ATTNPGM)
3
LOOP:
   CALL       QCMDEXC (&CMD 512)
   MONMSG     MSGID(CPF2320) /* F3 from
              Pgmrs Menu */
   TFRGRPJOB GRPJOB(*PRV)
   GOTO       LOOP
   ENDPGM
```

Figure 6-8. Initial Program for Group Jobs

The Attention key handling program (ATTNPGM) would be coded as follows:

```
PGM
TFRGRPJOB  GRPJOB(*SELECT)
MONMSG     MSGID(CPF1310) EXEC(DO)
           /* No group jobs */
SNDUSRMSG  MSG('An attention program
           is active, but no +
           group jobs exist') MSGTYPE(*INFO)
ENDDO      /* No group jobs */
ENDPGM
```

If you press the Attention key, and then press F6 to start a new group job (one that is not in your standard list), you must enter the SETATNPGM command in the new group job for the Attention key to be active.

If you want to end all group jobs, you can do so by calling the following program:

```
         PGM
         DCL        &GRPJOBL *CHAR LEN(1056)
         DCL        &X *DEC LEN(5 0) VALUE(67)
                    /* 2nd group job */
         RTVGRPA    GRPJOBL(&GRPJOBL)
         MONMSG     MSGID(CPF1311) EXEC(DO)
                    /* Not a group job */
         SNDPGMMSG  MSG('The current job is not a
                    group job')
         RETURN
         ENDDO      /* Not a group job */
LOOP:    IF         (&X *NE 1057) DO /* Less
                    than 16 group jobs */
         IF         (%SST(&GRPJOBL &X 10) *NE '    ')
                    DO /* Job */

         ENDGRPJOB  GRPJOB(%SST(&GRPJOBL &X 10))
         ENDDO      /* Job */
         CHGVAR     &X (&X + 66)
         GOTO       LOOP
         ENDDO      /* Less than 16 group jobs */
         CHGGRPA    GRPJOB(*NONE)
         SNDPGMMSG  MSG('All group jobs ended
                    and the + current job is no
                    longer a group job')

         ENDPGM
```

The program retrieves the current group job list, which can be made up of 16 entries where each entry is 66 bytes long containing:

Group job name	10 characters
Job number	6 characters
Group job text	50 characters

The active group job is the first one in the list. The program starts by trying to access the name of the second group job (starts in position 67) and if it is not blank, it ended the group job. When all other group jobs are ended, the current job is changed into a nongroup job.

Returning to the Group Job Main Program: In all previous examples, the work station user was allowed to go to any function from any other function. An alternative is to have the work station user always return to the main program. This could be done using the previous example program and removing the SETATNPGM command, as follows:

```
         PGM
         OVRDBF . . . SHARE(*YES)
         OPNDBF . . .
LOOP:    CALL  CUSINQ
         TFRGRPJOB  GRPJOB(*PRV)
         GOTO  LOOP
         ENDPGM
```

The work station user cannot use the Attention key. Ending the function (exiting from the CUSINQ program) returns the work station user to the main program.

As in the previous alternative, the only way to end this group job is to use the ENDGRPJOB command, use the ENDJOB command, or to sign off.

Note: Using the OVRDBF and OPNDBF commands to specify a shared open, allows a faster open each time a group job is called.

Transferring to Another Group Job without Seeing a Menu: You can use the Attention key to transfer directly to another job without seeing a menu. For example, the Attention key handling program for group job A could transfer to group job B. The Attention key handling program for group job B could transfer back to group job A. This allows a single keystroke to be used to switch between functions.

Ending Group Jobs

In some environments it may be desirable to force the end user to correctly end certain group jobs rather than issuing the ENDGRPJOB command. For example, assume that the user may have a group job where there is a complex update involved and you want to be sure the job is ended normally. Another example is where the user may be in the middle of a SEU session and should complete the function normally.

It is possible to achieve this with the support given by the system. For example, you could do the following:

- Set a switch in the group data area that could be tested by each of the group jobs to function as the shutdown switch. That is, when the switch is set on, the group jobs function should be ended.
- Access the active group job names by using the RTVGRPA command and the GRPJOBL return variable. Compare each name accessed (start with the second group job) against a predetermined list of the group job names that should be correctly ended. If the group job name is not in the list, it can be ended immediately by the ENDGRPJOB command. If the job must be correctly ended,

transfer to the group job using the TFRGRPJOB command.

- The Attention key handling program for all group jobs would have to be sensitive to the shutdown switch and would prevent transferring to another group job if the switch is set on.
- If you have a controlling program for each of the group jobs that controls what happens when the user ends the function of the group job (for example, the update program), it could also test the shutdown switch and do a return. This ends the group job and returns control to the previous active group job.
- The Attention key handling program can use the CHKRCDLCK command to determine if the work station user pressed the Attention key when the application had a record locked for update. In this case, the attention program may send a message instructing the user to complete the operation before using the Attention key.

Chapter 7. Batch Jobs

Jobs that do not require user interaction to run can be processed as batch jobs. A batch job is typically a low priority job and can require a special system environment in which to run. Batch jobs can be started when a user:

- Causes a job to be placed in a job queue
- Issues a communication program start request
- Starts a subsystem with an autostart job entry
- Starts a subsystem with a prestart job

Job Queues

You can think of a **job queue** as a list of batch jobs waiting to be processed. Actually, a job queue is an index of status and work-related information associated with a specific kind of work to be performed by the system. This work, referred to as a job on the AS/400 system, waits on a queue until the queue is allocated to an active subsystem. Once allocated, work information is retrieved from the queue one job at a time and used to start each job in the subsystem.

You can create job queues for groups of jobs and night jobs. The parameters on the Create Job Queue (CRTJOBQ) command specify (parameter names are given in parentheses):

- If a user having job control special authority can control the job queue and its contents (OPRCTL) (for example, if a job can be ended by someone other than the user who submitted the job)
- Authority (AUT)
- Text description (TEXT)

Once you create a job queue you must assign it to a subsystem by adding a job queue entry to a subsystem description. This is done with the Add Job Queue Entry (ADDJOBQE) command. The parameters on this command specify (parameter names are given in parentheses):

- Number of jobs that can be active at the same time on this job queue (MAXACT)
- In what order the subsystem handles work from this job queue (SEQNBR)
- How many jobs can be active at one time for each of nine levels of priority (MAXPTYn) (n=1 through 9)

A job queue can be associated with several subsystems but it can only be allocated to one subsystem at a time. Job queues are allocated to a subsystem in one of two ways. When the subsystem is started, the subsystem monitor tries to allocate each job queue defined in the subsystem job queue entries. If a job queue was allocated by another subsystem already, the first subsystem must end and deallocate the job queue before the second subsystem can allocate it. After it is started, this second subsystem allocates job queues assigned to it as they become available.

If a job queue does not exist when the subsystem is started, the job queue is allocated to the subsystem when:

- The job queue is created.
- A job queue is renamed with the name defined to the subsystem.
- A job queue is moved to another library and the resulting qualified name matches the name in the subsystem description.
- The library containing the job queue is renamed and the resulting qualified name matches the name in the subsystem description.

How Jobs Get on a Job Queue

You can place a job on a job queue by entering any one of the following commands:

- Submit Database Jobs (SBMDBJOB): Read input from a single-record physical or logical file.
- Submit Diskette Jobs (SBMDKTJOB): Read an input stream from a diskette device.
- Submit Job (SBMJOB): Use one job to place another job on a job queue.
- Start Database Reader (STRDBRDR): Read a batch input stream from a database and place one or more jobs on job queues.

- Start Diskette Reader (STRDKTRDR): Start a spooling reader to a diskette unit, read an input stream, and place it on a job queue.
- Transfer Job (TFRJOB): Move this job to another job queue in an active subsystem.
- Transfer Batch Job (TFRBCHJOB): Move this job to another job queue.
- Add Job Schedule Entry (ADDJOBSCDE): Automatically the system submits a job to the job queue at the time and date specified in the job schedule entry.

How Jobs Are Taken from Multiple Job Queues

A subsystem can have more than one job queue entry and can therefore allocate more than one job queue. The sequence number (SEQNBR) parameter of the Add Job Queue Entry (ADDJOBQE) command is used to specify the order in which the job queues are processed by the subsystem. The subsystem processes jobs from the job queue with the lowest sequence number first. When all jobs on that job queue have been processed, or when the maximum number of jobs from the queue is reached, the subsystem processes jobs from the queue with the next higher sequence number. The maximum number of jobs from a queue is specified by the MAXACT parameter on the Add Job Queue Entry (ADDJOBQE) or the Change Job Queue Entry (CHGJOBQE) commands.

The sequence continues until the subsystem has processed all available job queue entries or until the subsystem has reached its limit of jobs that can be running or waiting in the subsystem. The number of jobs that can be running or waiting is determined by the Maximum Jobs (MAXACT) parameter in the subsystem description. In some cases the sequence is interrupted as jobs end or are transferred. Creating, holding, and releasing job queues can also change the sequence of job queues processed.

The following is an example of how a subsystem handles jobs on several job queues:

Job Queue A (SEQNBR = 10)
Job 1
Job 2
Job 3

Job Queue B (SEQNBR = 20)

Job 4
Job 5
Job 6

Job Queue C (SEQNBR = 30)

Job 7
Job 8
Job 9

Each job queue entry in this example is specified as MAXACT(*NOMAX). The subsystem first selects jobs from job queue A because the job queue entry has the lowest sequence number. If the maximum number of jobs in the subsystem is 3 (MAXJOBS(3) parameter on the CRTSBSD command), it can select all the jobs from job queue A to be active at the same time. When any of the three jobs is completed, the activity level is no longer at the maximum; therefore a new job is selected from job queue B because it has the next lowest sequence number (assuming no new jobs have been added to job queue A).

Because each job queue entry specifies MAXACT(*NOMAX), an unlimited number of jobs can be started. Had each job queue entry specified MAXACT(1), then jobs 1, 4, and 7 would have been started. Had job queue entry A been specified as MAXACT(2), then jobs 1, 2, and 4 would have been started.

Input and Output Spooling

Often when jobs are processed by a subsystem, the rate that data can be handled by an input or output device is different than the rate that the system can handle it. This is obvious on a printer for example. It takes more time for a printer to print the data than for the system to get the data from a database file. Also, when a system has several users doing similar jobs, each user would like to use the system as if it were dedicated to his or her job. The system uses spooling to help in both of these situations.

Spooling allows the system to store data in an object called a spooled file. The spooled file collects data from a device until a program or device is available to process the data. A program uses

a spooled file as if it were reading from or writing to an actual device. This is input and output spooling.

Input spooling is done by the system for database and diskette files. An IBM-supplied program, called a reader, is started in the spooling subsystem, reads the batch job streams from the device, and places the jobs on a job queue.

Output spooling is done for printers. An IBM-supplied program, called a printer writer, is started in the spooling subsystem, selects spooled files from its output queue, and writes the records in the spooled output file to the printer.

Spooling support is available for two types of queues:

- Job queues. Entries for jobs are placed on job queues by readers, by the Submit Database Jobs (SBMDBJOB) command, by the Submit Diskette Jobs (SBMDKTJOB) command, by the Submit Job (SBMJOB) command, or by transferring to a subsystem using the Transfer Job (TFRJOB) or Transfer Batch Job (TFRBCHJOB) command. A subsystem selects jobs from a job queue for running.
- Output queues. Entries for the spooled output files made by jobs are placed on an output queue. A writer selects the output from the queue for writing to an output device.

Both the job queue name and the default output queue name for a job are specified in the job description for the job. Both can be overridden using parameters on the BCHJOB and SBMJOB commands. In addition, the output queue specified for a job already on a job queue or active in the system can be changed for files not yet opened by the job by using the Change Job (CHGJOB) command. Job queue entries in a subsystem description specify from which job queues a subsystem is to receive jobs.

Job Initiation

This section tells you how to initiate a batch job.

Batch Job Illustrations

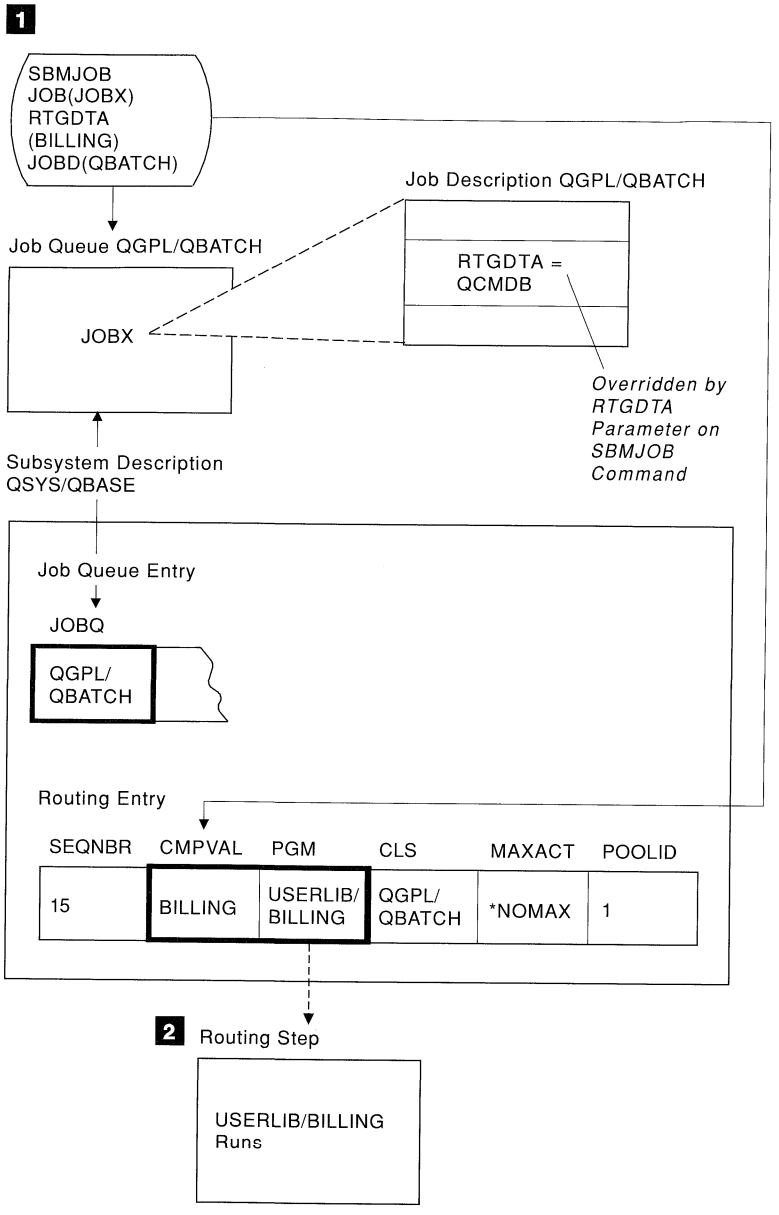
Figure 7-1 on page 7-4 and Figure 7-2 on page 7-5 show two ways of routing batch jobs.

- Using a user program to control the routing step for a job submitted using the SBMJOB command
- Using the IBM-supplied program QSYS/QCMD to control the routing step of a job submitted through a spooling reader

In each illustration, the user program USERLIB/BILLING is called.

Note: If no library is specified on the program start request, the subsystem library list is used to find the program.

Figure 7-1 on page 7-4 shows a batch job submitted through the SMBJOB command.



1 An entry for JOBX has been placed on the QGPL/QBATCH job queue by using a SBMJOB command. (The SBMJOB command can be entered interactively as shown or can be in an input stream.)

The routing data in the job description QGPL/QBATCH is overridden by the RTGDTA parameter specified on the SBMJOB command **1**. (All other parameters on the SBMJOB command are not specified and default to the values in the job description QGPL/QBATCH or use values from the job that is running at the same time.)

The routing data specified (BILLING) matches the comparison value in the routing entry specifying program USERLIB/BILLING.

2 USERLIB/BILLING is called for the routing step.

RSLS866-4

Figure 7-1. Batch Jobs Using the SBMJOB Command

Figure 7-2 on page 7-5 shows a batch job submitted through a spooling reader.

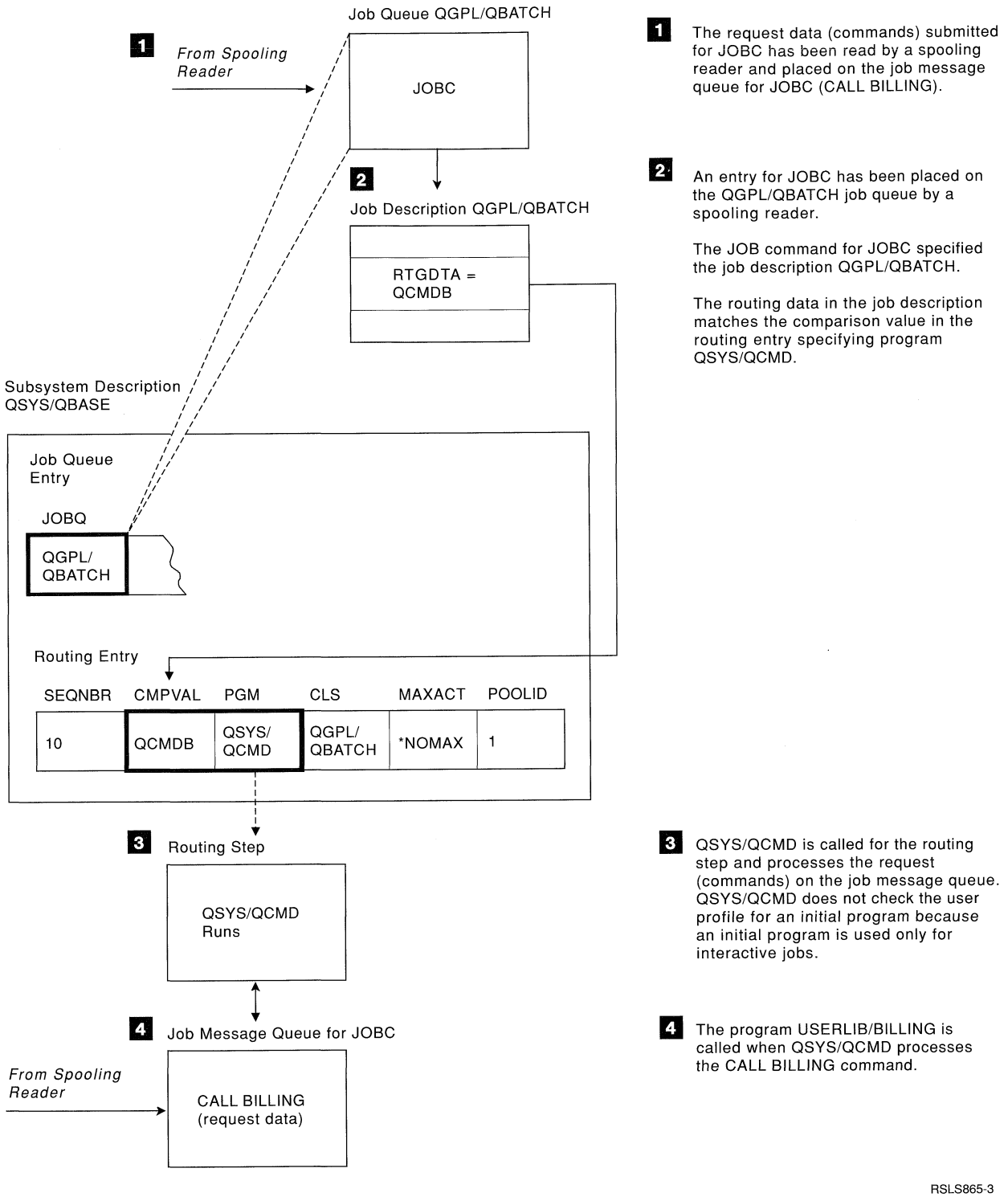


Figure 7-2. Batch Jobs Submitted through a Spooling Reader

An entry for JOBC was placed on the job queue QGPL/QBATCH by a reader. The entry on the job queue contains a description of the job and scheduling information. The request data and any inline data are not included in the job queue entry. The reader might have read the job from diskettes or a database file. This particular job contains only request data consisting of commands. If the job had contained any inline data files, the reader would have spooled them so they could be used during job running. The input stream in the database file was:

```
//BCHJOB JOB(JOBC) JOBD(QBATCH) JOBQ(QBATCH)
CALL BILLING
//ENDBCHJOB
```

The source for the request data is determined from the RQSDTA parameter on the BCHJOB command. The RQSDTA parameter can specify any one of the following:

- The actual request data (a character string for the parameter value)
- That the request data is in the job description (parameter value of *JOBD)
- That the request data follows the BCHJOB command (parameter value of *)

In Figure 7-1 on page 7-4, the RQSDTA parameter is not specified; it defaults to *, which means that the request data follows the BCHJOB command.

All other parameters on the BCHJOB command are not specified and default to the values in the job description QGPL/QBATCH.

Batch Job Considerations

The preceding batch job illustrations show the basic ways in which batch work can be handled. The primary considerations in determining the best approach for a particular job are:

- Running QSYS/QCMD to process a stream of commands gives both the functional capabilities of CL and the flexibility of dynamically changing the CL source before submitting the job.
- Running QSYS/QCMD as the routing step program allows processing of a single command, for example, issuing the CALL command to call an application program. The called program may contain commands (including other CALL commands) to do the functions of an application. In most environments where changes are minimal, this approach offers the greatest flexibility with minimal system resource.
- Running your application program as the routing step program minimizes the amount of interpretive command processing. This is not normally significant, but may be worth considering if jobs are started very frequently and performance is critical.

How To's

This section tells you how to perform some common tasks involving batch jobs.

Sending Completion Messages for Each Batch Job

You can use the MSGQ parameter on the BCHJOB or Submit Job (SBMJOB) command to specify the name of the message queue to which a completion message is to be sent when the job has completed (message CPF1241 indicates a normal completion and CPF1240 indicates an abnormal completion). For example, to send a completion message to the QSYSOPR message queue, you could use the following BCHJOB command:

```
//BCHJOB JOB(PAYROLL) JOBD(PAYROLL) MSGQ(QSYSOPR)
```

Instead of sending the completion messages for all jobs to the QSYSOPR message queue, you may want to send the completion messages for some jobs to the message queue of the submitting work station and for other jobs to a message queue that is periodically displayed or printed.

Another alternative is to send all the completion messages to a user message queue. You can then use a program to receive the messages from this queue and to resend them if necessary. For example, if a job failed for a user who is not a programmer or the system operator, the program could resend the message to both the work station user and the system operator.

Submitting Batch Jobs

The SBMDBJOB and SBMDKTJOB commands perform the same functions as readers in that they interpret input streams from diskette(s) or a database file and place jobs on job queues.

The SBMDBJOB and SBMDKTJOB commands:

- Run in the same job as the requester of the command. (The work station is locked until the submit jobs command has completed running.)
- Do not perform any syntax checking of the jobs in the input stream.

- Require fewer resources than a reader because a separate job is not started.
- Process the input stream sooner than a reader.

Whereas a reader:

- Runs in its own job. (Once the command has been specified, you can continue with other work.)
- Performs syntax checking on any job in the input stream in which syntax checking is specified on the BCHJOB command or in the job description.
- Requires more system resources than the submit jobs commands because a separate job is created.

In general, you may prefer to use the SBMXXXJOB commands unless the input stream is large or unless you want to use the syntax checking function done by a reader.

The Submit Job (SBMJOB) command and the submit jobs (SBMDBJOB and SBMDKTJOB) commands all can be used to submit batch jobs to a batch job queue. The difference in these commands is where the job comes from:

- The SBMJOB command can be used to submit a job to a batch job queue by specifying a job description and by specifying a CL command or request data, or specifying routing data to run a program.
- If you want to run a single CL command in the batch job, the simplest way is to use the CMD parameter on SBMJOB, which does syntax checking and allows prompting.
- The SBMDBJOB and SBMDKTJOB commands can be used to submit a job to a batch job queue from a diskette device, or a database file. For these jobs, the job description comes from the BCHJOB statement in the input stream.

For a detailed description of these commands, see the *CL Reference* manual.

In the following example, the SBMJOB command submits a job named WSYS, using the job description QBATCH, to the job queue QBATCH. The CMD parameter gives the CL command that will run in the job.

```
SBMJOB  JOBD(QBATCH)  JOB(WSYS)  JOBQ(QBATCH)
        CMD(WRKSYS)
```

Note: If you get a message that the job was not submitted, you can display the job log spooled file to find errors. Use the WRKJOB command. Specify the job that was not scheduled, select option 4 for spooled files. Display the job log spooled file to find the errors.

Scheduling a Batch Job

The job schedule function allows for time-dependent scheduling of AS/400 batch jobs.

- You can control the time a job is submitted to the job queue by defining a job schedule entry.

  To define a job schedule entry, use the Add Job Schedule Entry (ADDJOBSCDE) command. The system automatically submits a job to the job queue at the specified date and time.
- You can control the time a job is released in the job queue by using the Submit Job (SBMJOB) command and specifying values other than the defaults for SCDDATE and SCDTIME parameters.

For more information, see Chapter 11, "Job Scheduling" on page 11-1.

Submitting a Job Using Parameters from a Display File

As an example of how you can submit a batch job from another subsystem, the following describes a CL program that prompts for input from a work station user and passes the input received as parameters to a batch job. A display file is used to prompt the user.

The following is the display file (SBMJOBSMPD), which is used to prompt the user for the parameters to be passed:

```
SEQNBR *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7
  1.00      A* SBMJOBSMPD - Submit data from display - used by SBMJOBSMPC
  2.00      A          R PROMPT
  3.00      A                                        CF03(93 'RETURN')
  4.00      A                                     1  2'Prompt for SMBJOB'
  5.00      A                                     3  2'Name'
  6.00      A            NAME          10   I    +2
  7.00      A                                     5  2'Department'
  8.00      A            DEPT           3   I    +2
  9.00      A                                     7  2'Days'
 10.00      A            DAYS           3  0I    +2
 11.00      A                                     9  2'Amount'
 12.00      A            AMOUNT         7  2I    +2
```

The following is the CL program (SBMJOBSMPC), which is used to display the SBMJOBSMPD file and submit the job:

```
SEQNBR *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7
  1.00 /* SBMJOBSMPC - Submit job to batch with parameters                */
  2.00              PGM
  3.00              DCLF       FILE(SBMJOBSMPD)
  4.00              DCL        &DAYSA *CHAR LEN(3)
  5.00              DCL        &AMOUNTA *CHAR LEN(8)
  6.00              SNDRCVF
  7.00              IF         (&IN93 *EQ '1') RETURN
  8.00              CHGVAR     &DAYSA &DAYS
  9.00              CHGVAR     &AMOUNTA &AMOUNT
 10.00              SBMJOB     JOB(JOBA) JOBD(JOBA) RQSDTA('CALL SBMJOBSMP +
 11.00                           PARM(' *CAT &NAME *BCAT '''' *CAT +
 12.00                           &DEPT *BCAT '''' *BCAT &DAYSA *BCAT +
 13.00                           &AMOUNTA *CAT ')') SWS('00000001')
 14.00              ENDPGM
```

The program prompts for the parameters and receives data from the display file. The data is concatenated together to form the CMD parameter on the SBMJOB command.

The data from the display requires special processing because of the following rules:

- When you concatenate CL variables to make a parameter value, the CL variables must be character variables. You must also specify blanks, apostrophes, and parentheses that are to be included in the parameter to be passed.
- If you specify a parameter value starting with a digit (such as 43T) on a CALL statement, the system interprets the parameter as a numeric parameter.
- When a parameter is specified within apostrophes, and you want to specify another apostrophe within it, you must specify two apostrophes.

If the following information was specified on the prompt:

```
NAME      JONES
DEPT      57A
DAYS      5
AMOUNT    5000
```

the command submitted to the batch job queue would be:

```
CALL SBMJOBSMP PARM(JONES '57A' 5 5000)
```

Data is received from the display in the following fields:

- NAME: The work station user should key in all alphabetic data (no leading digits) for this field. It is received as a character variable in the CL program, and no special processing is required for the data.
- DEPT: The work station user could start with a digit when typing in the DEPT field (for example, 43T). If 43T is passed to the

system on a parameter in a command, the system would treat the parameter as a numeric variable. This is the wrong data type; the DEPT parameter must be character data.

To prevent this error, you must enclose the department number in apostrophes (such as '43T'). However, in the CL program, you must specify four apostrophes. When the CL program is compiled, the four apostrophes are processed by the system and become two apostrophes. When the CL program is run, the two remaining apostrophes are processed by the system again, and become one apostrophe.

- DAYS and AMOUNT: These numeric fields must be converted to character data so they can be concatenated into the RQSDTA parameter on the SBMJOB command. The program declares two character variables (DAYSA and AMOUNTA), then changes their value to the value of the numeric variables (DAYS and AMOUNT) as received from the display file. The program uses the character variables for the concatenation.

If the following information was specified on the prompt:

```
NAME      JONES
DEPT      43T
DAYS      2
AMOUNT    931
```

the request data sent to the batch job queue would be:

```
CALL SBMJOBSMP PARM(JONES '43T' 002 009.31)
```

The AMOUNT field contains a decimal point because it was changed from the numeric field AMOUNT to a character field by the CHGVAR command. The character variable must be long enough to contain this data.

An example of coding the SBMJOBSMP program to receive the parameters in batch (if it were a CL program) is shown in this example:

```
SEQNBR *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ..

  1.00              PGM        PARM(&NAME &DEPT &DAYS &AMOUNT)
  2.00              DCL        &NAME *CHAR LEN(10)
  3.00              DCL        &DEPT *CHAR LEN(3)
  4.00              DCL        &DAYS *DEC LEN(15 5)
  5.00              DCL        &AMOUNT *DEC LEN(15 5)
  6.00              DCL        &DAYSD *DEC LEN(3 0)
  7.00              DCL        &AMOUNTD *DEC LEN(7 2)
  8.00              CHGVAR     &DAYSD &DAYS
  9.00              CHGVAR     &AMOUNTD &AMOUNT
 10.00             .
 11.00             .
```

The program receives the character parameters with the desired definition. The numeric parameters must be received with a definition of LEN(15 5). (All numeric variables passed in from a processing program are passed as 15 5.) The data is then moved to the desired length fields by the CHGVAR command. If both parameters of the CHGVAR command are decimal variables, decimal alignment is performed.

An example of coding the SBMJOBSMP program to receive the parameters in batch (if it were an RPG program) is shown in this example:

```
SEQNBR *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ..

  1.00      C           *ENTRY    PLIST
  2.00      C                     PARM           NAME   10
  3.00      C                     PARM           DEPT    3
  4.00      C                     PARM           DAYS   155
  5.00      C                     PARM           AMOUNT 155
  6.00      C                     Z-ADDDAYS      DAYSD   30
  7.00      C                     Z-ADDAMOUNT    AMTD    72
  8.00      .
  9.00      .
```

The program uses the same type of definition as the CL program to receive the parameters. The Z-ADD operation is used to convert the numeric data to the desired field definitions because it performs decimal alignment.

Creating Job Queues

Use the Create Job Queue (CRTJOBQ) command to create a job queue on the system. The following example creates a job queue called JOBQA in the LIBA library:

```
CRTJOBQ JOBQ(LIBA/JOBQA) TEXT('test job queue')
```

Some additional parameters are used to specify authority and control the job queue. If these are not included, you, as owner of the job queue, can control the use of the job queue. Also, those with access to the library can use the job queue.

Allocating Job Queues

Once you create a job queue, you also need to associate it with one or more subsystems. The subsystems can process work from the job queue. Use the Add Job Queue Entry (ADDJOBQE) command to associate a job queue to a subsystem. The following example adds a job queue entry for the JOBQA job queue in the TEST subsystem description. There is no maximum number of jobs that can be active on this job queue and the work is processed with a sequence number of five.

```
ADDJOBQE SBSD(TEST) JOBQ(LIBA/JOBQA)
         MAXACT(*NOMAX) SEQNBR(5)
```

Finding a Job in a Job Queue

Use the Work with Job Queues (WRKJOBQ) command to find a job in a job queue. Enter the following command to see a list of all jobs on the JOBQA job queue:

```
WRKJOBQ JOBQ(LIBA/JOBQA)
```

If you do not know the name of the job queue, enter the command without the JOBQ parameter. The Work with All Job Queues display appears with a list of all job queues to which you are authorized. Scan this list until you see the name of a job queue that may contain the job you are trying to find.

Looking at Jobs in a Job Queue

Once you have found a job in a job queue (see "Finding a Job in a Job Queue"), you can look at that job by entering the work with option for the job you want to see. The Work with Job display appears. This display provides several options for viewing all information available for the job you selected.

How many jobs can be on a job queue? There is no set limit to the number of jobs that can be on a job queue. It is limited only by how much work the system can handle.

Changing the Number of Jobs Running from a Job Queue

The QBASE subsystem is shipped with a job queue entry for the QBATCH job queue that only allows one batch job at a time to be running. If you want to allow more batch jobs from that job queue to be running at the same time, use the Change Job Queue Entry (CHGJOBQE) command. The following command would allow

two batch jobs from the QBATCH job queue to be running at the same time in the QBASE subsystem. (This command can be issued at any time and takes effect immediately.)

```
CHGJOBQE SBSD(QBASE) JOBQ(QBATCH) MAXACT(2)
```

Determining Which Subsystem Has a Job Queue Allocated

You may not be able to submit a job to a job queue because it is allocated to another subsystem. To determine which subsystem has the job queue allocated (JOBQA), enter the following command:

```
WRKJOBQ JOBQ(LIBA/JOBQA)
```

The Work with Job Queue display appears. The subsystem description function key appears in the function key area of the display when the job queue is allocated to a subsystem. Press the subsystem description function key and the Work with Subsystem Descriptions display appears showing the subsystem to which the job queue is allocated.

Determining Which Job Queues are Allocated to a Subsystem

You may want to see if there are any jobs waiting to be processed by a subsystem. To see this you must see if there are any job queues allocated to that subsystem. Enter the WRKJOBQ command and the Work with All Job Queues display appears with all the job queues you are authorized to on the system. This display also shows the subsystem running on the system that have allocated the job queues listed and the number of jobs in each job queue.

Seeing Which Job Queues are Associated with a Subsystem

Use the Display Subsystem Description (DSPSBSD) command to see which job queues are associated with a subsystem. For example:

```
DSPSBSD SBSD(QBASE)
```

The Display Subsystem Description menu appears. Select the Job Queue Entries option. The Display Job Queue Entries display appears with a list of all job queues associated with the QBASE subsystem.

Moving a Job from One Job Queue to Another

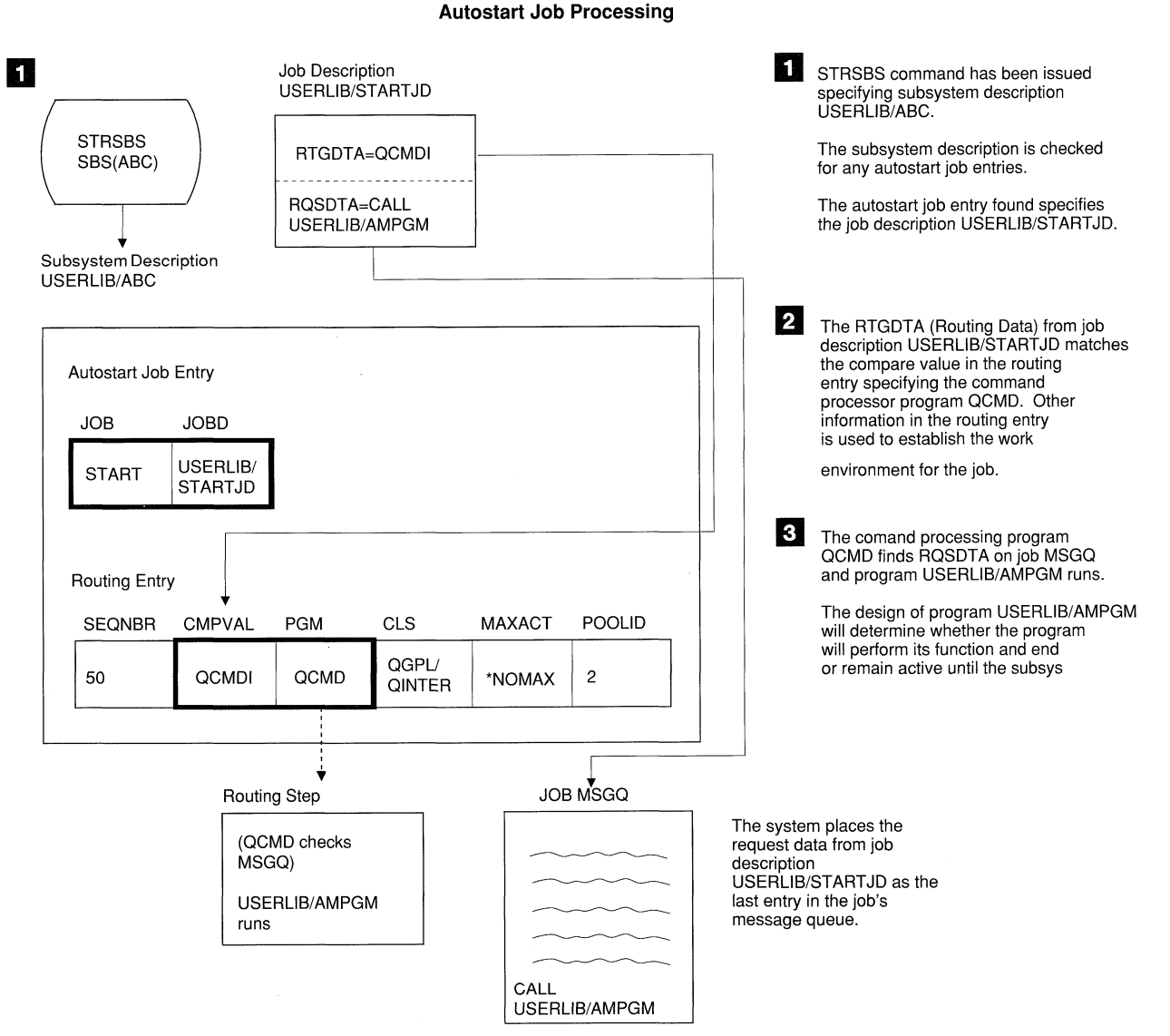
You can move a job you are authorized to from one job queue to another by using the Change Job (CHGJOB) command. The following example moves JOBA to JOBQB:

```
CHGJOB JOB(LIBA/JOBA) JOBQ(LIBA/JOBQB)
```

Chapter 8. Autostart Jobs

Using autostart jobs, you can automatically start jobs that perform repetitive work, initialize functions for an application, or provide centralized service functions for other jobs in the same subsystem. An autostart job in the controlling subsystem can be used to bring up other subsystems (as does the IBM-supplied controlling subsystem).

Figure 8-1 illustrates autostart jobs.



Note: If more than one autostart job is specified for a subsystem, all autostart jobs are started immediately, not one followed by another. If the maximum number of jobs of the subsystem is exceeded, no other jobs can be started in the subsystem until enough autostart jobs have completed so that the number of jobs running is below the maximum activity level.

RV2W790-0

Figure 8-1. Autostart Job Initiation

Initiating Jobs

Autostart jobs are associated with a particular subsystem and each time the subsystem is started, the autostart jobs associated with it are started. An autostart job entry in a subsystem description specifies a job that is to be automatically started when a subsystem is started. To add an autostart job entry to the subsystem description, use the Add Autostart Job Entry (ADDAJE) command.

Security Considerations

The job description used for an autostart job is specified using the ADDAJE command. When the subsystem is started, the job operates under the user profile name in the specified job description. You may not specify the job description USER(*RQD). Because the job description is located by qualified name at the time the subsystem starts, you may want to control access to the job description.

Chapter 9. Communications Jobs

Job processing involves a communication request and appropriate specifications.

Initiating Jobs

For a communications batch job to run on an AS/400 system, a subsystem description containing a work entry for communications jobs must exist on the system. The communications work entry identifies to the subsystem the sources for the communications job it will process. The job processing begins when the subsystem receives a communications program start request from a remote system and an appropriate routing entry is found for the request.

Routing Data for Communications Jobs

Job routing of communications jobs is determined by the program start request that is received from the remote system. When a program start request is processed on the target system, a fixed-length data stream that is used as routing data is created. Position 29 of the routing data will always contain PGMEVOKE for communications requests. Subsystem routing entries that specify a compare value of PGMEVOKE in position 29 typically have *RTGDTA as the program name. This means that the program name specified in the routing data (from the remote system's program start request) is the program to run.

If a special processing environment is required for certain communications jobs, you can add an additional routing entry to the subsystem description, specifying a compare value whose starting position is 37. This compare value should contain the program name for the program start request. The routing entry must have a sequence number lower than the routing entry that uses PGMEVOKE as the compare value. This method allows certain communications jobs to run with a different class and/or pool specification.
Figure 9-1 on page 9-2 shows an example of a program start request and the associated routing entry for such a job.

Security Considerations

The security on the AS/400 system controls who can use communications devices as well as who can access the commands used with the associated device descriptions. You should consider additional security measures when writing and running application programs on both remote and target systems.

The job description used for communications jobs is specified on the Add Communications Entry (ADDCMNE) command. The user specified on this job description is ignored. The system gets the user name for communications jobs from the program start request. If the program start request does not specify a user name, the system uses the default user value from the communications entry. To ensure a greater degree of system security, include user information on the program start request rather than specifying a default user in the communications work entry.

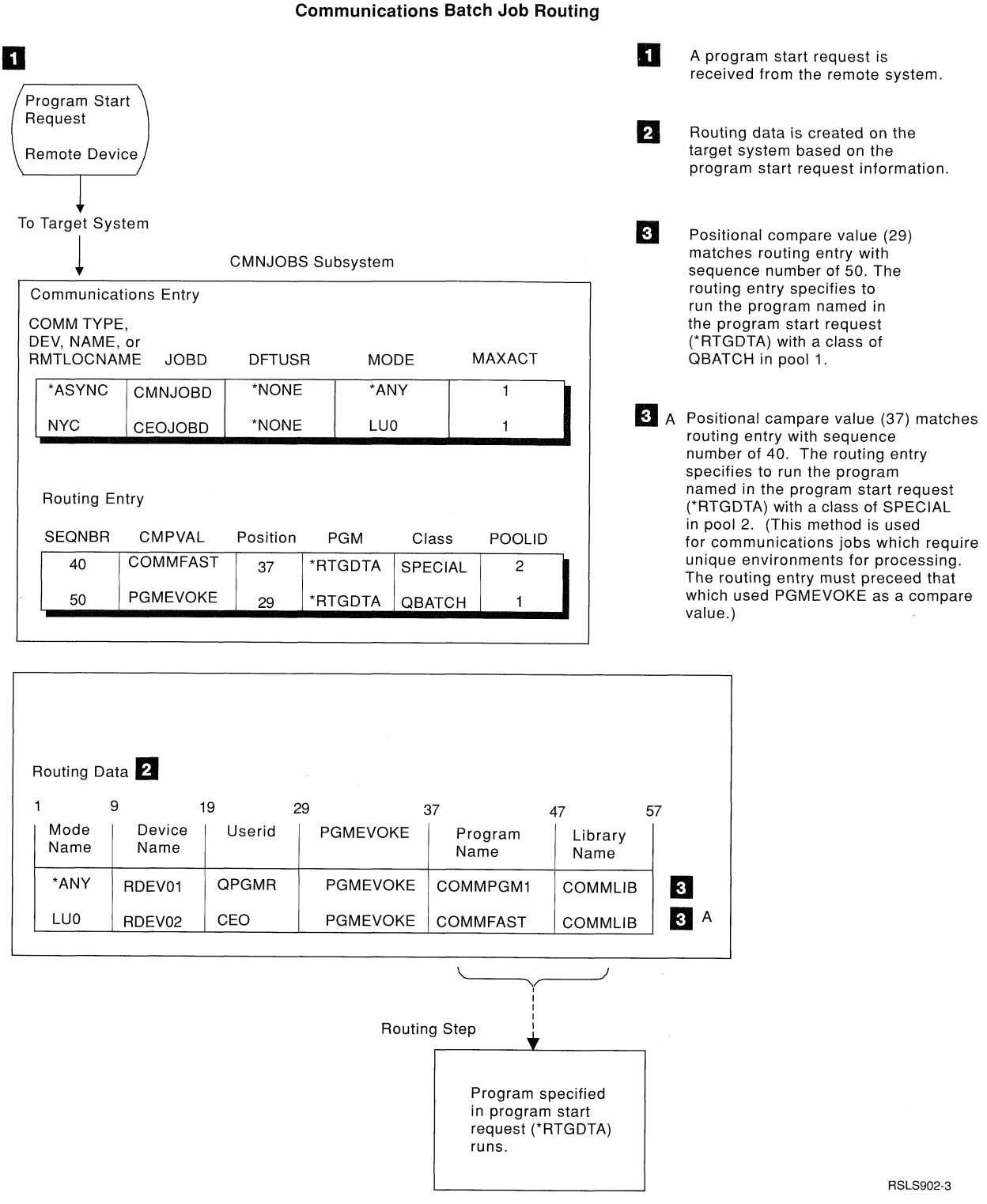


Figure 9-1. Communications Batch Job Routing

Note: If the library is not specified on the program start request, the subsystem library list is used to find the program.

Chapter 10. Prestart Jobs

This chapter describes how to use prestart jobs to reduce the amount of time required to handle a program start request. Included is information on how to configure, start, handle, and end prestart jobs.

Initiating Jobs

Prestart jobs are different from other jobs because they use prestart job entries to determine which program, class, and storage pool to use when they are started. When a subsystem is started, or when the Start Prestart Job (STRPJ) command is entered, prestart jobs are started based on the information contained in the prestart job entries. When a program start request is received on the target system, it is sent to the subsystem that has the required communications device allocated.

The program start request attaches to a prestart job that is already running if the subsystem finds either of the following:

- A prestart job entry with a program name that matches the program name of a program start request
- A routing entry that matches the routing data of the program start request **and** the routing program on the found routing entry matches the program name on a prestart job entry

If the prestart job entry is not active, the program start request is rejected. If a match is not found, the program start request causes a communications batch job to start.

A prestart job entry does not affect the device allocation or program start request assignment. The prestart jobs are started before any other types of jobs in a subsystem.

To use prestart jobs, you must define a prestart job entry and a communications entry in the subsystem description. Figure 10-1 on page 10-2 shows prestart job initiation.

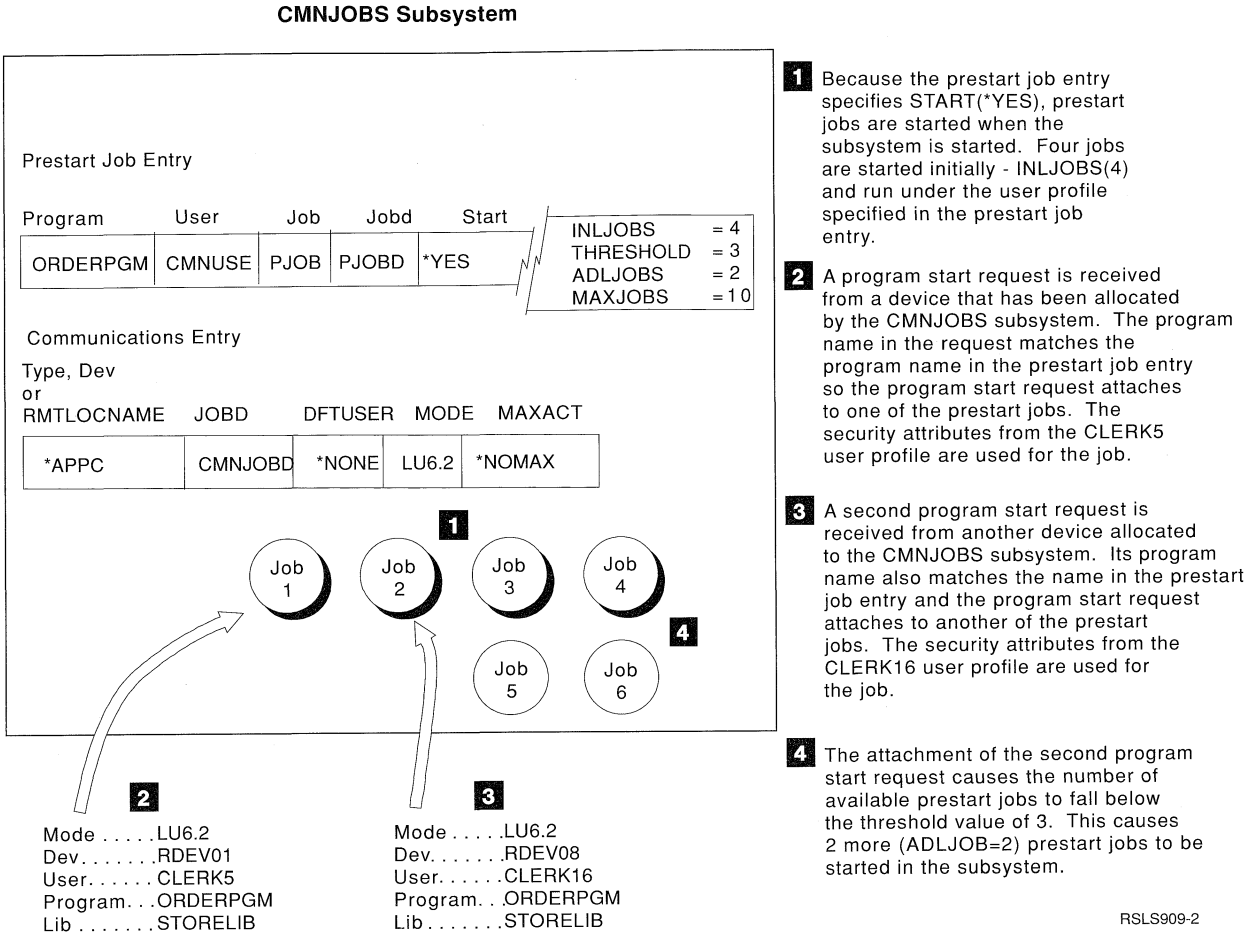


Figure 10-1. Prestart Job Initiation

Defining a Prestart Job

Within a prestart job entry, you must specify attributes that the subsystem uses to create and manage a pool of prestart jobs. The following is a description of each attribute in a prestart job entry (beginning with those attributes unique to a prestart job entry):

Attributes Unique to A Prestart Job Entry

Initial number of jobs (INLJOBS):
The initial number of prestart jobs you want to start when the prestart job entry starts.

Threshold (THRESHOLD):
Specifies when you want additional prestart jobs to start. When the number of prestart jobs that are available to service a program start request are reduced below this value, more jobs (the number specified in the additional number of jobs attribute) start.

Additional number of jobs (ADLJOBS):
The additional number of prestart jobs that you want to start when the number of prestart jobs drops below the threshold value.

Maximum number of jobs (MAXJOBS):
The maximum number of prestart jobs that can be active at the same time for this entry. This value includes prestart jobs that are servicing a program start request, jobs that are waiting to service a program start request, and jobs that are being started as a result of reaching the threshold value.

Maximum number of program start requests (MAXUSE):
The maximum number of program start requests that can be handled by each prestart job before the job is ended by the subsystem.

Wait (WAIT):
Specifies whether program start requests wait for a prestart job to become available or are rejected if a prestart job is not immediately available when a program start request is received.

Start jobs (STRJOBS):
Specifies whether the prestart jobs should start when the subsystem starts. If you change this value when the subsystem is active, the change takes effect after the subsystem is ended and started again.

Program name (PGM):
The name of the program that the prestart job runs. This name will be used to match an incoming program start request with an available prestart job. (If you do not give a library qualifier, *LIBL is used to find the program.) If the program does not exist when the entry is added, you must specify a library qualifier because the qualified program name is retained in the subsystem description.

Note: Two entries with the same program name can exist in a single subsystem description as long as you specify different library names for the programs.

Pool identifier (POOLID):
The subsystem pool identifier in which the prestart job runs.

Class name (CLS):
The name of the classes under which the prestart jobs run and the number of prestart jobs that should run using each class. Jobs will start using the first class specified until the number of jobs specified for the first class is reached. After the number of jobs specified for the first class is reached, the jobs will start using the second class. If the class does not exist when you add the entry, you must specify a library qualifier because the qualified class name is retained in the subsystem description.

Attributes Common to Some Work Entries

Subsystem description (SBSD):
The name of the subsystem description to which the prestart job is added. (If you do not give a library qualifier, *LIBL is used to find the subsystem description.)

User profile (USER):
The user profile under which the prestart job runs when the prestart job is not servicing a program start request.

Job name (JOB):
The simple name of the prestart job that you want to start.

Job description (JOBD):
The name of the job description that you want to use for the prestart job. If the job description does not exist when you add the entry, you must specify a library qualifier, because the job description name is retained in the subsystem description.

Prestart Job Management

This section tells you how to perform some common tasks involving prestart jobs.

Number of Prestart Jobs

When the subsystem starts, the INLJOBS attribute on the prestart job entry determines the number of prestart jobs that initially start for a program. If more jobs need to be created, the ADLJOBS value is used. The THRESHOLD value is the least number of jobs that can be available before new jobs are started. There are some restrictions for the values of these parameters.

- INLJOBS must be > or = to THRESHOLD
- INLJOBS must be < or = to MAXJOBS
- ADLJOBS must be < MAXJOBS

If you specify *NO on the STRJOBS attribute, no prestart job starts for the prestart job entry when the subsystem starts. You can start the prestart jobs by using the STRPJ command when the subsystem is active. However, running the STRPJ command does not cause the value of the STRJOBS parameter to change.

The number of prestart jobs that can be active at the same time is limited by the MAXJOBS attribute on the prestart job entry and by the MAXJOBS attribute for the subsystem. The MAXACT attribute on the communications entry controls the number of program start requests that can be serviced through the communications entry at the same time.

Queuing Program Start Requests

If a program start request arrives when the current number of prestart jobs is less than the number specified in the MAXJOBS attribute on the prestart job entry, and none of the prestart jobs are available to handle program start request, you have the option to have this new request rejected or queued. This option is controlled by the WAIT attribute on the prestart job entry.

WAIT (*NO) means the following:

- If no prestart job is available immediately, the program start request is rejected.

WAIT (*YES) means the following:

- If no prestart job is available immediately and no prestart job can be started due to MAXJOBS to service the program start request, the program start request is rejected.
- If no prestart job is available immediately, but additional prestart jobs can be or have been started, the program start request is queued.

Controlling Prestart Jobs

If an active prestart job entry is in the subsystem, the subsystem periodically checks the number of prestart jobs in a pool that are ready to service program start requests to determine if there are excessive available prestart jobs. Excessive available prestart jobs are ended by the subsystem gradually. However, the subsystem always leaves at least the number of prestart jobs specified in the INLJOBS attribute in a pool.

Each time a program start request is received, the subsystem checks the number of prestart jobs in the pool that are not attached to a program start request. If the number drops below the minimum THRESHOLD value, the subsystem starts the number of prestart jobs specified in the ADLJOBS attribute. However, the subsystem does not start more than the number specified in the MAXJOBS attribute on the prestart job entry or the number specified in the MAXJOBS attribute for the subsystem.

If a prestart job goes to the end of the job either normally or abnormally (except if it is ended by the ENDJOB command) before one program start request attaches to the prestart job, the prestart job program is considered to be in error and the subsystem ends the prestart job entry.

If a prestart job goes to the end of the job after at least one program start request has been attached to it, the subsystem starts one prestart job to replace it (as long as the MAXJOBS value has not been reached).

Security Considerations

This section tells you how to perform some common security tasks on your system.

User Profile

When a prestart job starts, it runs under the prestart job user profile. When a program start request attaches to a prestart job, the prestart job user profile is replaced by the program start request user profile. When the prestart job is finished handling a program start request, the program start request user profile is replaced by the prestart job user profile. If there is a group profile associated with the user profile, the group profile is also exchanged. The exchange of the user profile is for authority checking only. None of the other attributes associated with the user profile are exchanged. Libraries, on the library list that the prestart job entry user profile is authorized to, continue to be authorized to the prestart job when the program start request user profile replaces the prestart job entry user profile. However, the library list can be changed by the Change Library List (CHGLIBL) command.

Object Authorization

When a prestart job starts, authority checking against the prestart job entry user profile is performed on every object that is needed for starting a job. Before a program start request is allowed to attach to a prestart job, only the program start request user profile/password and its authority to the communications device and library/program is checked. To avoid occurrences where the program start request user profile is not authorized to objects that the prestart job entry user profile is authorized to, you must ensure that the program start request user profile is authorized to at least as many objects as the prestart job entry user profile. To accomplish this, the prestart job program can be created by the prestart job entry user with USRPRF(*OWNER) specified on the CRTxxxPGM (where xxx is the program language) command. The program owner authority will automatically be transferred to any programs called by the prestart job program. Otherwise, you may choose to explicitly check object authorization (CHKOBJ) before referring to any objects.

Files and objects that the prestart job user profile is not authorized to should be closed and deallocated before end of transaction is performed on the requestor device. If database files are left open in the prestart job, the prestart job program must check the program start request user profile authority to the open files, in order to guarantee the database security.

Application Considerations

The prestart job should do as much work as possible before it attempts to acquire an ICF program device or accept a CPI Communications conversation. The more work it does initially (allocating objects, opening database files, and so on), the less it will have to do when a program start request is received, therefore giving the transaction faster response time.

You should only deallocate resources specific to the transaction you want performed. Any resource that is commonly used for other transactions performed by the prestart job program should remain allocated while the job is waiting for its next request. You should leave files open and objects allocated to save time when the next request is received.

Note: Database files that are left open in the prestart job generally require the same considerations as database files that are shared in the same job. For information on sharing database files between jobs and sharing database files in the same job, see the *Database Guide*.

Since the same QTEMP library is used for the entire life of a prestart job, objects that are no longer needed should be deleted.

Since the same Local Data Area (LDA) is used for the entire life of a prestart job, information can be kept and passed to the next transaction.

Since each prestart job can handle many program start requests, and has only one job log, you may want your application to send messages to the job log identifying the activity of the prestart job.

The job attributes of a prestart job are not changed by the subsystem when a program start request attaches to a prestart job. The Change Prestart Job (CHGPJ) command allows the prestart job to change some of the job attributes to those of the job description (specified in the job description associated with the user profile of the program start request or in the job description specified in the prestart job entry).

If your system uses job accounting, the prestart job program should run the CHGPJ command with the program start request parameter (CHGPJ ACGCDE(*PGMSTRRQS)) immediately after the program start request attaches to the prestart job. This action changes the accounting code to the value specified in the user profile associated with the program start request. Immediately before the program finishes handling the program start request, the program should run the Change Prestart Job command with the Prestart Job Entry parameter (CHGPJ ACGCDE(*PJE)). This changes the accounting code back to the value specified in the job description of the prestart job entry.

Once a prestart job is started, the three-part name of the prestart job never changes. The middle name of the three-part job name always contains the user profile under which the prestart job is started. It may not be the current user profile of the job.

If a spool file is opened before a prestart job handles any program start request, the spool file is associated with the prestart job entry user profile; otherwise it is associated with the current program start request user profile.

If the prestart job entry profile and the current program start request user profile are different, spooled files are spooled under a job with the first part of the three-part job name being QPRTJOB and the second part being the name of the user profile.

The class (CLS) parameter on the prestart job entry provides a way to control the performance characteristics of two classes of prestart jobs per prestart job entry. For example, the first class may specify attribute PURGE(*NO) and the second class may specify PURGE(*YES). This allows the class 1 jobs to attempt to stay in main storage when waiting for a program start request.

See the *ICF Programmer's Guide* for more information about setting up prestart job programs that use ICF. See the *APPC Programmer's Guide* for more information about setting up prestart job pro-

grams that use Common Programming Interface (CPI) Communications.

How To's

This section tells you how to perform some common tasks involving prestart jobs.

Starting and Ending Prestart Jobs

Use the following commands to start and end prestart jobs:

- To start jobs for a prestart job entry in an active subsystem, use the Start Prestart Jobs (STRPJ) command. Prestart jobs can also start at the same time the subsystem is started.
- To end all jobs for a prestart job entry in an active subsystem, use the End Prestart Job (ENDPJ) command.

Chapter 11. Job Scheduling

This chapter describes how to schedule batch jobs. Job scheduling allows you to control the date and time a job is submitted to or becomes eligible to start from a job queue. This flexibility can help you as you balance the work load on your system.

You can schedule a batch job by (1) defining a job schedule entry or by (2) using the Submit Job (SBMJOB) command. The characteristics of each job make one of the scheduling methods more effective for that job.

The SBMJOB command is an easy way to schedule a job that only needs to run once. It also allows you to use many of the job attributes defined for your current job. However, since the SBMJOB command places the job on the job queue immediately, you will lose your scheduled job if the job queue is cleared before the scheduled date and time.

Using a job schedule entry, you can schedule jobs to recur or to run only once. You may want to schedule a job to recur if it is a repetitious task, such as calculating payroll, filling out timecards, or sending out meeting notices. Since job schedule entries are entries in a permanent object, they do not stay on the job queue like the scheduled jobs, and therefore they are not lost when the job queue is cleared. You can also save and restore the job schedule object, providing a method of backing up your job scheduling information.

Scheduling a Job Using the Submit Job (SBMJOB) Command

Schedule a job using the Submit Job (SBMJOB) command by specifying a date and time on the SCDDATE and SCDTIME parameters. When you issue this command with those parameters specified, the job remains on the job queue in a scheduled state (SCD status) until the date and time indicated by the parameters.

At the scheduled time, the job is released from the job queue. The job's status changes from scheduled (SCD) to released (RLS), unless the job is held (SCD HLD), in which case it changes from scheduled to held (HLD). The job is then processed like any other job on the job queue. The normal conditions (such as a job queue allocated to an active subsystem and maximum jobs not already active) are required for the job to start.

Scheduled jobs (*SCD) do not necessarily start in the order they appear on the job queue when they are all scheduled to run at the same time. To make sure they start in a specific order, vary the scheduled starting time of each job by a minute.

Note: No message is issued to indicate the job's status has changed.

Clearing a Job Queue with a SBMJOB Scheduled Job: When a job queue with a scheduled job is cleared, the scheduled job is handled similarly to any other job on the job queue. The job is removed from the job queue and a message indicating that the job ended abnormally is issued to the message queue specified on the SBMJOB command. If the deleted job needs to remain scheduled, the program that submitted the job should monitor this message queue and resubmit the job if it receives the message.

Scheduling a Job Using a Job Schedule Entry

Another way to schedule a job is to add an entry to the job schedule object. Then, at the scheduled time, a job is submitted to the job queue. At this point, the only indication that the job came from the job schedule entry is that the job name in the *Submitted by* field on the Display Job Status Attributes display is the name of the job schedule (QJOBSCD) system job. You can view this display using the Work with Jobs (WRKJOB) command.

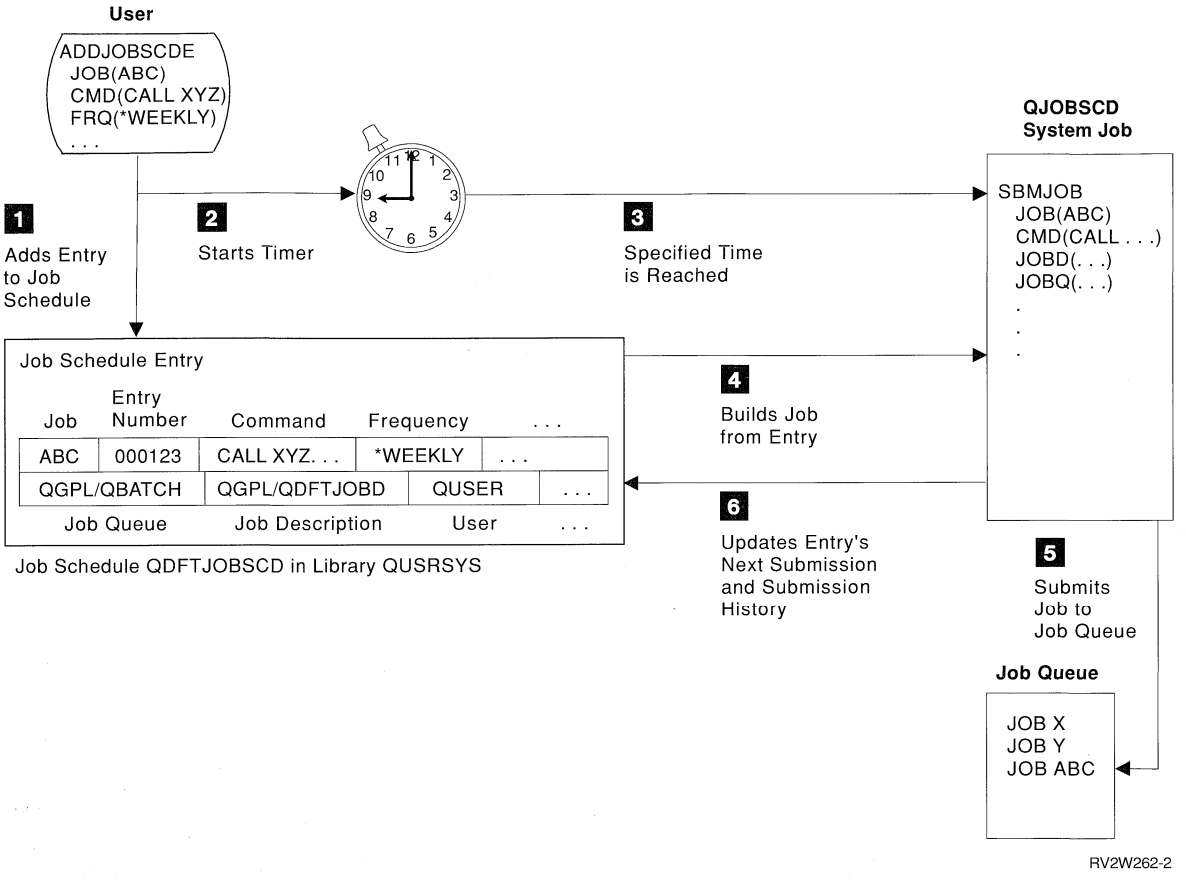


Figure 11-1. Job Schedule Entry Overview

Job Schedule Entry Commands

You can add, change, hold, release, and remove a job schedule entry. The following table lists the commands for these tasks:

Figure 11-2. Job Schedule Entry Commands

Command	Description
ADDJOBSCDE	Add a job schedule entry to the job schedule.
CHGJOBSCDE	Change a job schedule entry. This command changes the entry in the job schedule, but it does not affect any jobs already submitted using this entry.
HLDJOBSCDE	Hold a job schedule entry. If an entry is held, no job is submitted at the scheduled time due to the held status.
RLSJOBSCDE	Release a job schedule entry. If the scheduled time has not yet been reached, the status is changed to scheduled (SCD). If the scheduled time passed while the entry was held, no jobs are submitted, and a warning message is sent to indicate that a job or jobs were missed.
RMVJOBSCDE	Remove a job schedule entry from the job schedule.

Job Schedule Object

The job schedule object, QDFTJOBSCD, is in the QUSRSYS library and has an object type of *JOBSCD. You cannot create, delete, rename, or duplicate the job schedule object, and you cannot move it to any other library.

The job schedule object can be saved with the Save Library (SAVLIB), Save Object (SAVOBJ), or

Save Changed Objects (SAVCHGOBJ) command and then restored with the Restore Library (RSTLIB) or Restore Object (RSTOBJ) command. Restoring the job schedule object causes the next submission date to be updated for each entry. You can restore the job schedule object to the system from which it was saved or to a different system, but you cannot restore it to a library other than QUSRSYS. If you restore the job schedule object to a different system, the job submission information is cleared in each entry.

Defining a Job Schedule Entry

To schedule a job by defining a job schedule entry, use the Add Job Schedule Entry (ADDJOBSCDE) command. This command adds an entry to the job schedule object.

Figure 11-1 on page 11-2 shows job schedule initiation. Each entry in the object is uniquely identified by the job name you supply and a 6-digit entry number assigned by the system. No two entries have the same job name and entry number combination.

The job schedule entry contains all the information necessary to submit a job and the scheduling information. It also contains information used by the system to manage the entry in certain situations. The information that defines the job is similar to the parameters specified on a Submit Job (SBMJOB) command, including job name, command, job description, job queue, user profile, and message queue. The local data area (LDA) of the job submitted from the job schedule entry is blank when the job starts.

The following are other parameters, unique to the ADDJOBSCDE command, that allow you to define scheduling and handling information:

Frequency (FRQ)

The *Frequency* prompt for a job schedule entry defines how often the job will be submitted. The possible frequencies are

If you specify *MONTHLY for the *Frequency* and a date containing the 29th, 30th, or 31st day of the month for the *Schedule date*, a warning message is issued when you add or change the entry to indicate that a job will not be submitted from the entry every month. For example, if 01/31/99 is specified for the schedule date when *MONTHLY is specified for the *Frequency*, the job is submitted only on 01/31, 03/31, 05/31, 07/31, 08/31, 10/31 and 12/31. To make sure that a job runs on the last day of every month, *MONTHEND should be specified for the *Schedule date* prompt.

Schedule Date (SCDDATE)

The *Schedule date* prompt defines the date that the job is scheduled to be submitted. This date can be the first day of the month (*MONTHSTR), the last day of the month (*MONTHEND), or a specific date.

Either the *Schedule date* or the *Schedule day* should be used to define an entry, but not both.

Schedule Day (SCDDAY)

The *Schedule day* prompt defines the day of the week that the job is scheduled to be submitted. It accepts the values *ALL, *NONE, *MON, *TUE, *WED, *THU, *FRI, *SAT, and *SUN.

Multiple schedule days, up to a maximum of seven, can be specified to control the days of the week on which a job is submitted. For example, you can schedule a job to be submitted daily (Monday through Sunday), or just on the weekdays (Monday through Friday).

Either the *Schedule date* or the *Schedule day* should be used to define an entry, but not both.

Schedule Time (SCDTIME)

The *Schedule time* prompt defines the time on the scheduled date (or scheduled day) when the job is to be submitted.

Although the precision for the schedule time can be specified to the second, the activity involved in submitting the job and the current load on the system may affect the exact time at which a job is submitted.

As a result, job schedule entries are not necessarily submitted in the order they appear. To make sure they are submitted in a specific order, vary the scheduled starting time of each job schedule entry by a minute.

Relative Day of the Month (RELDAYMON)

The *Relative day of the month* prompt indicates which day of the week, relative to the time of the month, the job should be submitted. For example, if *MON is specified for the schedule day and **2** is specified for the rel-

ative day of the month, the job is scheduled to run on the second Monday of each month.

Omit Date (OMITDATE)

The *Omit date* prompt defines certain dates (up to 20) when the job should not run. You can control the specific dates on which a job should not run for a recurring entry without repeatedly changing the entry.

Recovery Action (RCYACN)

The *Recovery action* prompt defines what happens if the job cannot be submitted at the scheduled time because the system is powered down or in restricted state. The recovery action occurs during the system IPL or when the system comes out of restricted state. Through the recovery action you can indicate whether to submit the job in a released state, *SBMRLS; to submit the job in a held state, *SBMHLD; or to not submit the job, *NOSBM.

This *Recovery action* prompt does not define action when a job was held at the scheduled time and then released at a later time. Also, the recovery action does not define action when scheduled times are passed as a result of changes to the QTIME system value.

When the missed jobs are submitted during an IPL, they are submitted in the order in which they would have been submitted if the system had not been down at the scheduled dates and times. The first missed occurrence of a recurring job is used to order the jobs.

Note: Since the job schedule portion of IPL need not be complete for the IPL of the system to be complete, other jobs could start on the system before all of the jobs have been submitted.

Save entry (SAVE)

The *Save entry* prompt specifies whether an entry that has FRQ(*ONCE) should be kept in the job schedule object after the job has been submitted.

Although you may have scheduled a job to run only once, you may want to save it for your records or for future use. If *ONCE is specified for the *Frequency* prompt and *NO is specified for the *Save* (SAVE) prompt, the entry is removed from the job schedule object when the job is submitted. If, however, *YES is specified for the *Save* prompt when *ONCE is specified for the *Frequency*, the entry remains in the job schedule as a saved entry and the entry remains inactive until the CHGJOBSCDE command is used. If *ONCE is not specified, the system updates the date and time in the job schedule entry.

Note: If you specify to save the entry (*YES), no jobs are submitted again for that entry after the initial job is submitted. To schedule a job again using the entry, use the Change Job Schedule Entry (CHGJOBSCDE) command to specify a new date and time.

The following parameter values are also used when the job is submitted. These values are not default values and you do not specify them in the job schedule entry:

Print device	*JOBD
Output queue	*JOBD
Print text	*JOBD
System library list	*SYSVAL
Current library	*USRPRF
Initial library list	*JOBD
Coded character set ID	*USRPRF
Language ID	*USRPRF
Country ID	*USRPRF
Sort sequence	*USRPRF

Printing a List of Scheduled Jobs

To print a list of job schedule entries, use the Work with Job Schedule Entries (WRKJOBSCDE) command:

- For a list of jobs:

```
WRKJOBSCDE OUTPUT(*PRINT)
```

- For detailed information on each job schedule entry:

```
WRKJOBSCDE OUTPUT(*PRINT) PRTFMT(*FULL)
```

To print a list of jobs that were scheduled on the Submit Job (SBMJOB) command:

```
WRKSBSJOB SBS*(JOBQ) OUTPUT(*PRINT)
```

This command gives you a list of all the jobs on job queues; it doesn't separate those that are scheduled from those that are not.

Calendar Considerations

If you have a calendar that is not Gregorian, take note of the following:

Restoring the job schedule object: Use extreme caution when restoring the job schedule object to a system that has a different calendar.

You must properly change each entry to ensure that the scheduling information is correct for the current calendar. Failing to update the entries may cause unpredictable job scheduling.

Other Considerations:

- Supporting days of the week

  The support for days of the week (Monday through Sunday) may not work correctly if the system is using a non-Gregorian calendar. Use of these values on a non-Gregorian calendar is not prohibited, but the scheduling occurs according to the Gregorian calendar for those entries.

- Changing the QLEAPADJ system value

  Changing the QLEAPADJ system value may cause unpredictable job scheduling unless you update the entries to ensure that the scheduling information is correct or you restore the job schedule object.

- Using dates that are not valid

  When you use a date that does not exist (such as 2/29 in a year that is not a leap year), you receive a message instructing you to correct that entry. Jobs are not submitted from an entry if the date is not valid.

Working with Job Schedule Entries

The Work with Job Schedule Entries (WRKJOBSCDE) command allows you to work with the entries in the job schedule. You can select a subset of the list by the entry name, the scheduled submission date, the user profile that added the entry, or the job queue to which the job will be submitted.

```
                        Work with Job Schedule Entries              RCHASLOG
                                                          11/22/92  09:54:58

Type options, press Enter.
  2=Change     3=Hold   4=Remove  5=Display details   6=Release
  8=Work with last submission     10=Submit immediately

                                                                      Next
                         -----Schedule------              Recovery   Submit
Opt  Job         Status  Date       Time      Frequency    Action    Date
5    PAYROLL      SCD    11/19/92   15:48:46  *MONTHLY     *SBMRLS   12/19/92




                                                                      Bottom
Parameters or command
===>
F3=Exit   F4=Prompt          F5=Refresh   F6=Add     F9=Retrieve
F11=Display job queue data   F12=Cancel   F17=Top    F18=Bottom
```

From the Work with Job Schedule Entries display, you can add (F6), change (option 2), hold (option 3), remove (option 4), and release (option 6) job schedule entries. You can also display the details of an entry (option 5), work with the last job successfully submitted (option 8) or submit a batch job using the information contained in an entry (option 10).

Display Details: Selecting option `5=Display details` from the Work with Job Schedule Entries display presents the Display Job Schedule Entry Details display. This display provides more information about a specific job schedule entry, including the command to be run and the submission history of the entry.

```
                        Display Job Schedule Entry Details
                                                                System:   RCHASLOG
Job:   PAYROLL        Entry number:   000062     Status:   SCD

Last attempted submission:
  Status . . . . . :   Job successfully submitted.

  Date . . . . . :   11/19/92
  Time . . . . . :   15:48:46
Last successful submission:
  Job  . . . . . :   PAYROLL
    User . . . . . :     PAYROLLMGR
    Number . . . . . :   011870
  Date . . . . . :   11/19/92
  Time . . . . . :   15:48:46
Schedule date  . . . . . :   11/19/92
Schedule time  . . . . . :   15:48:46
Frequency  . . . . . :   *MONTHLY
Recovery action  . . . . . :   *SBMRLS
                                                                        More...
Press Enter to continue.

F3=Exit   F6=Display last successful submission   F12=Cancel
```

Submission history: The submission history provides information that is useful in determining what has happened with a particular entry. The history information is divided into two sections: last attempted submission and last successful submission.

- Last attempted submission.

  Last attempted submission information shows what happened the last time the system was able to submit a job using this entry. The status indicates if the job was successfully submitted, if the submission attempt failed, or if the job was not submitted for another reason.

- Last successful submission.

  Last successful submission information shows the last time a job was successfully submitted. In addition to the date and time of the successful submission, the fully qualified name of the submitted job is shown. If a job was successfully submitted, you can use `F6=Display`

`last successful submission` from this display to see the Display Job menu for the submitted job.

Work with last submission: Selecting option `8=Work with last submission` presents the Work with Job menu for the last job successfully submitted using this entry. This is a quicker way to find the job listed in the *Last successful submission* section of the Display Job Schedule Entry Details display.

Submit immediately: Selecting option `10=Submit immediately` presents the Submit Job (SBMJOB) command prompt display with job schedule entry information shown for the command's parameters. You can change any of the parameter values and then press the Enter key to submit the job to a job queue. Using this option does not change any information in the job schedule entry, including the last submission history and the next submission date and time.

Security Considerations

The job schedule object QDFTJOBSCD in library QUSRSYS is shipped with public authority of *CHANGE. This is the minimum authority necessary to add, change, hold, release, and remove job schedule entries.

Adding a Job Schedule Entry: To add a job schedule entry, you must have *CHANGE authority to the job schedule object and *USE authority to the user profile. In addition, you and the user profile must have the following authorities to the objects referenced on the Add Job Schedule Entry command:

- *USE authority to the job description
- *READ authority to the job queue
- *USE and *ADD authority to the message queue
- *READ authority to all the libraries associated with all these objects

Changing a Job Schedule Entry: To change a job schedule entry, you must have the same authorities that are required to add an entry. However, the authorities to the individual objects are checked only if you are changing that parameter for the entry. In addition, if you do not have *JOBCTL special authority, you can change only the entries that your user profile added to the job schedule object.

Holding, Releasing, and Removing a Job Schedule Entry: When holding, releasing, or removing a job schedule entry, you must have *CHANGE authority to the job schedule object. If you do not have *JOBCTL special authority, you can hold, release, or remove only the entries that your user profile added to the job schedule object.

Working with Job Schedule Entries: To work with the job schedule entries, you must have *USE authority to the job schedule object. In addition, if you want to display the details of an entry, either with option `5=Display details` or the *FULL print format (PRTFMT parameter), you must have *JOBCTL special authority. Otherwise, you can display the details for only the entries that your user profile added to the job schedule object.

List Job Schedule Entries (QWCLSCDE) API: To use the SCDL0100 format of the List Job Schedule Entries API, you must have *USE authority to the job schedule object. To use the SCDL0200 format, you must have *JOBCTL special authority in addition to *USE authority to the job schedule object. If you do not have *JOBCTL special authority, you cannot see all the information available for the job schedule entries.

Recovery Considerations

System Availability: If the system is powered down or in restricted state when scheduled times are reached, jobs cannot be submitted from job schedule entries and the status of scheduled jobs cannot be changed. However, the system handles this situation after the system IPL or after it comes out of restricted state.

Using **job schedule entries**, you can control how each entry is handled by the value you specify for the recovery action of the entry. You can specify that a job still be submitted using the entry, that a job be submitted and held on the job queue, or that a job should not be submitted. If you request that a job be submitted, only one job is submitted from each entry, no matter how many submissions were missed while the system was not available.

Using **scheduled jobs**, the system checks to determine if any scheduled times have passed

while the system was not available. If a scheduled job with a passed time is found, the job's status is updated as described in "Scheduling a Job Using the Submit Job (SBMJOB) Command" on page 11-1.

Note: The job schedule entries and the scheduled jobs are processed in the order that the missed occurrences would have been handled normally. However, this is not required to complete before the system becomes available again. Therefore, work from other sources may enter the system while missed job schedule entries and scheduled jobs are being processed.

Damaged Job Schedule Object: If an attempt to access a damaged job schedule is made, the object is deleted and an empty job schedule is created. If the job schedule object is damaged, you can restore the job schedule object during normal system operations. The restore operation does not require that the system be taken to restricted state. When the object is restored, the next occurrence for each job schedule entry is calculated and normal job scheduling starts again.

How To's

The following section shows examples of adding job schedule entries for a variety of jobs, an example program using system messages, and the effects of changing the QDATE and QTIME system values.

Examples of Adding a Job Schedule Entry

The following examples show how the job schedule function can help manage a wide range of tasks.

Monthly: This example shows how to submit a job to run program INVENTORY at 11:30 p.m. on the last day of every month except on New Year's Eve.

```
ADDJOBSCDE JOB(MONTHEND)
           CMD(CALL INVENTORY)
           SCDDATE(*MONTHEND)
           SCDTIME('23:30:00')
           FRQ(*MONTHLY)
           OMITDATE('12/31/93')
```

Daily: This example shows how to submit a job to run program DAILYCLEAN every day at 6:00 p.m. The job runs under the user profile SOMEPGMR. This job is not submitted if the system is down or is in restricted state at that time.

```
ADDJOBSCDE JOB(*JOBD)
           CMD(CALL DAILYCLEAN)
           SCDDAY(*ALL) SCDTIME('18:00:00')
           SCDDATE(*NONE)
           USER(SOMEPGMR)
           FRQ(*WEEKLY) RCYACN(*NOSBM)
```

Weekly: This example shows how to submit a job to run program PGM1 every week starting on 12/17/93 at the current time. Because 12/17/93 is a Friday, the job is submitted every Friday, and it runs under the user profile PGMR1.

```
ADDJOBSCDE JOB(*JOBD)
           CMD(CALL PGM1)
           SCDDATE('12/17/93') FRQ(*WEEKLY)
           USER(PGMR1)
```

Third Monday and Wednesday: This example shows how to submit a job to run program PGM2 on the third Monday and the third Wednesday at 11:30 p.m. This job will be submitted on the next third Monday or third Wednesday at 11:30 p.m., depending on whether those days have passed already this month. If yesterday was the third Monday, today is the third Tuesday, and tomorrow is the third Wednesday, it will be submitted tomorrow, and then not again until next month.

```
ADDJOBSCDE JOB(*JOBD)
           CMD(CALL PGM2)
           SCDDAY(*MON *WED) FRQ(*MONTHLY)
           SCDDATE(*NONE)
           RELDAYMON(3) SCDTIME('23:30:00')
```

First and Third Monday: This example shows how to submit a job to run program PAYROLL on the first and third Monday of every month at 9:00 a.m. The job runs under user profile PAYROLLMGR.

```
ADDJOBSCDE JOB(PAYROLL)
           CMD(CALL PAYROLL)
           SCDDAY(*MON) FRQ(*MONTHLY)
           SCDDATE(*NONE)
           RELDAYMON(1 3) SCDTIME('09:00:00')
           USER(PAYROLLMGR)
```

Every Weekday: This example shows how to submit a job to run PGM4 every weekday at 7:00 p.m.

```
ADDJOBSCDE JOB(*JOBD)
           CMD(CALL PGM4)
           SCDDAY(*MON *TUE *WED *THU *FRI)
           SCDDATE(*NONE)
           SCDTIME('19:00:00') FRQ(*WEEKLY)
```

Save an Entry: This example shows how to submit a job once and save the entry.

```
ADDJOBSCDE   JOB(*JOBD)
             CMD(CALL SAVED)
             FRQ(*ONCE)
             SAVE(*YES)
```

Using Messages

You can use system messages to retrieve an entry number, monitor entry removal, or to generally monitor the status of your job scheduling.

Adding a Job Schedule Entry: When an entry is added, either CPC1238 or CPC1244 (added with warnings) is returned. The entry number assigned to the entry can be retrieved from this message.

Successfully Submitting a Job from a Job Schedule Entry: When the job schedule system job submits a job, either CPC1236 or CPC1242 (submitted with warnings) is sent to the message queue specified for the entry and to the QHST log to record the job submission. If the job schedule entry does not specify a message queue, CPC1243 is sent to the QSYSOPR message queue only.

Failing to Submit a Job from a Job Schedule Entry: When the system fails to submit a job, either CPI1119, CPI1120, or CPI1141 is sent to the message queue specified in the entry and to the QSYSOPR message queue in library QSYS to notify you of the failed job submission. If the job schedule entry does not specify a message queue, CPI1142 is sent to the QSYSOPR message queue only.

Removing a Job Schedule Entry: When the system job removes a single-submission entry or when an entry is removed by the Remove Job Schedule Entry (RMVJOBSCDE) command, CPC1239 is sent to the message queue specified in the entry. If a single-submission entry was held when its scheduled time was reached and the entry specified *NO for its save attribute, the entry is removed when it is released with the Release Job Schedule Entry command. In this case, CPC1245 is sent to the message queue specified in the entry.

The following example program shows how you can use system messages. The program retrieves the entry number from the message automatically and then uses it to change the entry.

```
PGM

    /* Declare program variables */
    DCL VAR(&ENTRYID) TYPE(*CHAR) LEN(6)
    DCL VAR(&MSGDATA) TYPE(*CHAR) LEN(50)
    DCL VAR(&MSGDLEN) TYPE(*DEC)  LEN(5 0) VALUE(50)
    DCL VAR(&MSGID)   TYPE(*CHAR) LEN(7)

    /* Issue Add Job Schedule Entry command */
    ADDJOBSCDE JOB(EXAMPLE) CMD(WRKDSKSTS +
               OUTPUT(*PRINT)) FRQ(*WEEKLY) +
               SCDDATE(*NONE) SCDDAY(*FRI) +
               SCDTIME('18:00:00')

    /* Check for function check */
    MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(ADDFAIL))

    /* Receive last message */
    RCVMSG     MSGTYPE(*LAST) RMV(*NO) MSGDTA(&MSGDATA) +
               MSGDTALEN(&MSGDLEN) MSGID(&MSGID)

    /* Check for correct completion message */
    IF         COND(&MSGID *NE 'CPC1238') +
               THEN(GOTO CMDLBL(ADDFAIL))

    /* Retrieve entry number from message data */
    CHGVAR     VAR(&ENTRYID) +
               VALUE(%SUBSTRING(&MSGDATA 31 6))

    /* Issue Change Job Schedule Entry command */
    CHGJOBSCDE JOB(EXAMPLE) ENTRYNBR(&ENTRYID) +
               SCDDAY(*MON *WED *FRI)

    /* Check for function check */
    MONMSG     MSGID(CPF0000) +
               EXEC(GOTO CMDLBL(CHGFAIL))

    /* Branch to end of program */
    GOTO       CMDLBL(ENDOFPGM)

ADDFAIL:
    /* Branch point for failure on ADDJOBSCDE command */
    SNDPGMMSG  MSG('Add Job Schedule Entry Failed')
    GOTO       CMDLBL(ENDOFPGM)

CHGFAIL:
    /* Branch point for failure on CHGJOBSCDE command */
    SNDPGMMSG  MSG('Change Job Schedule Entry Failed')

ENDOFPGM:
    /* End of program */
ENDPGM
```

Changing the QDATE or QTIME System Values

Changes to the QDATE or QTIME system values have the following effects on job scheduling:

Moving the date or time backward: If you move the date or time backward, the scheduled date and time of the job schedule entries are not calculated again. If you want the submission dates and times for the job schedule entries calculated after you move the date and time backward, you must restore the job schedule object.

Moving the date or time forward: If you move the date or time forward while the system is not in restricted state, the job schedule entries with scheduled times falling within the time change are submitted immediately and only once. The next submit times for the entries are calculated based on the current time.

If you move the date or time forward while the system is in restricted state, the job schedule entries with scheduled times falling within the time change are treated as if they had expired in restricted state. When the system comes out of restricted state, the recovery action is applied to the entries and the next submit time for the entries is calculated based on the current time.

If you do not want the jobs submitted when a date or time is moved forward, the entries should be held. Then, when the time is moved forward, you receive messages for the missed occurrences, and the next submit time for the entries is calculated based on the current time. You only need to release all the held entries for normal processing to resume.

Chapter 12. System Jobs

A **system job** is a job created by the OS/400 operating system. Work such as controlling system resources and scheduling jobs is performed by system jobs. System jobs can be divided into the following categories:

- Start-control-program-function (SCPF)
- System arbiter (QSYSARB)
- logical unit services (QLUS)
- Work control block table cleanup (QWCBTCLNUP)
- Performance adjustment (QPFRADJ)
- Database server (QDBSRV1..N)
- Decompress system object (QDCPOBJ1..N)
- Job schedule (QJOBSCD)
- System spool (QSPLMAINT)
- Alert manager (QALERT)
- Subsystem monitor
- System work subsystem (QSYSWRK)

Start-Control-Program-Function (SCPF)

The SCPF system job provides the environment and directs the functions necessary to start the OS/400 licensed program during an initial program load (IPL). Elements of the SCPF function call several module and non-module interfaces that perform functions such as checking for and displaying deferred program change prompts, displaying the SCPF sign-on prompt, and initiating the system arbiter process.

The SCPF job remains active (at a priority equal to batch jobs) after the operating system is started, providing an environment for the running of low-priority and possibly long-running system functions. The SCPF job also ends the machine processing after the arbiter ends.

System Arbiter (QSYSARB)

The system arbiter (QSYSARB), started by an SCPF system job, provides the environment for the running of high-priority functions. It allows subsystems to start and end and keeps track of the state of the system (for example, a restricted state).

The system arbiter, identified by the job name QSYSARB, is the central and highest priority job within the operating system. The system arbiter responds to system-wide events that must be handled immediately and those that can be handled more efficiently by a single job than multiple jobs.

The system arbiter is also responsible for starting the Logical Unit Services (QLUS) job during an IPL. The system arbiter remains active until the system is ended.

QLUS

The Logical Unit Services, identified by the job name QLUS, supports communications devices. It is started by the system arbiter, even if there are no communications devices on the system. QLUS handles the event handling for logical unit devices (communications devices) and also acts as the manager of communications devices.

QWCBTCLNUP

The work control block table (WCBT) cleanup system job is used during the initial program load (IPL) for WCBT cleanup. This system job completes before the end of the IPL.

QPFRADJ

The performance adjust system job manages changes to the storage pool sizes and activity levels. All requests to change storage pools are processed by this job. In addition, if system value QPFRADJ is set to '2' or '3,' this job will dynamically change the sizes and activity levels of storage pools to improve the performance of the system.

QDBSRV1..N

The database server system jobs perform database access path recovery during the IPL and normal system operation. These system jobs are started on every IPL and are active until the system is ended.

QDCPOBJ1..N

The decompress system object system jobs decompress compressed system objects when they are needed. These system jobs are started on every IPL.

QJOBSCD

The job schedule (QJOBSCD) system job controls the job scheduling functions of the system. This job monitors the timers for job schedule entries and scheduled jobs. This system job starts during IPL and remains active until the system is powered down.

QSPLMAINT

The system spool (QSPLMAINT) system job performs system spool functions. The job starts after the IPL and performs the following functions:

- If the status of the job is DLTSPLF, the system job clears the data from a spool database member after the user has specified to delete a spooled file.
- If the status of the job is RCLSPLSTG, the system job deletes spool database members in the QSPL library that have been unused and empty. Use system value QRCLSPLSTG to adjust the number of days to hold empty spool database files.
- If the status of the job is SPLCLNUP, the system job performs the spool cleanup operation.

  After an abnormal IPL:

  - Jobs are moved from destroyed job queues to QSPRCLJOBQ in QRCL.
  - Spooled files are moved from destroyed output queues QSPRCLOUTQ in library QRCL.
  - Spooled files on destroyed device output queues are moved to the recreated device output queues.

  When a damaged spooled database file is encountered, spooled files with data are deleted.

QALERT

The alert manager (QALERT) system job performs the tasks necessary to process alerts. This includes such activities as processing alerts received from other systems, processing locally created alerts, and maintaining the sphere of control. The alert manager job remains active until the system is ended.

Subsystem Monitor

The subsystem monitor job provides control over an active subsystem. It provides functions such as initiating, controlling, and ending of jobs. There may be several subsystem monitor jobs running on a system at any given time.

Subsystem monitor jobs are identified by type SBS on the Work with Active Jobs display. You can see this by using the Work with Active Jobs (WRKACTJOB) command.

Note: IBM supplies two complete controlling subsystem configurations, QBASE (the default controlling subsystem) and QCTL. Only one controlling subsystem can be active on the system at one time. The controlling subsystem is determined by the Control Subsystem Description (QCTLSBSD) system value. See "Detailed Description of System Values" on page 2-11 for additional information about system values. Also see Appendix C, "Characteristics of the Shipped System" for more information.

QSYSWRK

The system work subsystem (QSYSWRK) contains jobs that support system functions started automatically at IPL and when the system comes out of restricted state. It is possible to start and end this subsystem or use job commands, such as the Change Job (CHGJOB) command, to change the jobs in this subsystem. However,

changing these jobs affects the operation system functions. Use caution changing these jobs.

QECS Job: An example of a job running in QSYSWRK subsystem is QECS. QECS is a job that provides communications sessions between AS/400 systems when one of the systems is the service provider for others. (A system is a service provider when it has SystemView System Manager/400 installed and configured on the system. The other system or systems are called service requesters.) These communications sessions use SNA MS/Transport support.

The QECS job starts whenever the QSYSWRK subsystem starts. Thus, at every system IPL, the QECS job starts. The QECS job must be active on both AS/400 systems for the information about PTFs, service requests, and problem analysis to move between them.

System Jobs Illustration

1. The system is started.
2. The SCPF job starts the system arbiter job and then remains active as a background job until the arbiter ends.
3. The system arbiter job starts the logical unit services (QLUS) job to provide support for communications devices.
4. The system arbiter job calls device configuration initialization to allow vary-on processing.
5. The system arbiter starts the subsystem monitor job for the controlling subsystem and remains active until the system is ended.

Figure 12-1 illustrates how system jobs are started.

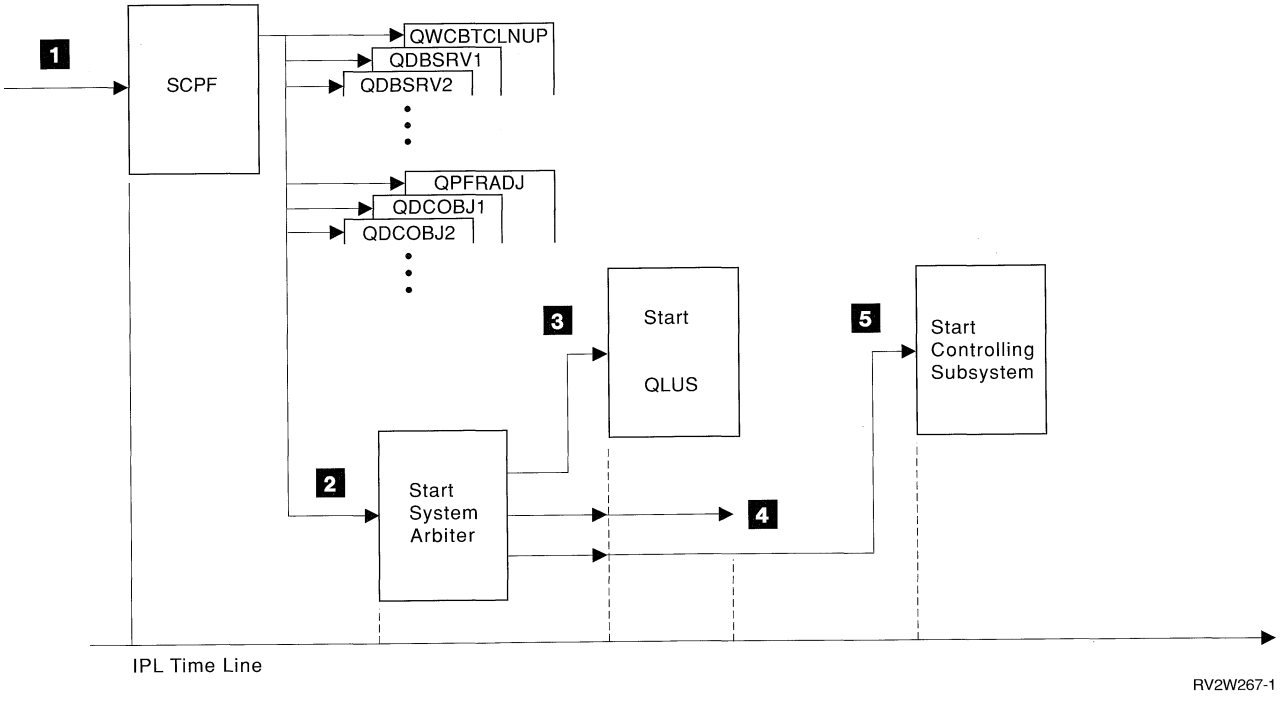


Figure 12-1. System Jobs Overview

How To's

This section tells you how to perform some common tasks involving system jobs.

Displaying Information about System Jobs

You can display information about system jobs by using the Work with Active Jobs (WRKACTJOB) command. After entering the command, the Work with Active Jobs display appears. Type a 5 in the option field next to the system job you want to display. Next, the Work with Job display appears. From this display, you can select to display information about the job. Selecting option 10 from this display shows the job log, which contains messages about the progress and status of the job. See "Job Logs" on page 4-8 for more information about job logs.

Chapter 13. Performance Tuning

This chapter describes how to make performance adjustments to your system. There are two approaches you can take when tuning the system:

- The system can make performance adjustments automatically.
- You can make the performance adjustments manually.

"Automatic System Tuning" discusses the approach that most users should take. For an expanded discussion of what occurs during automatic performance adjustment, see "Performance Adjustments" on page 13-18.

Most of the chapter discusses manual performance adjustment, including:

- Performance components
- Performance commands
  - Work with System Status (WRKSYSSTS)
  - Work with Active Jobs (WRKACTJOB)
  - Work with Disk Status (WRKDSKSTS)
- Basic tuning
- Specialized tuning

To make manual performance adjustments and understand the performance values that can be set, see "Performance Components" on page 13-2, "Performance Commands" on page 13-7, and "Basic Tuning" on page 13-11. If you are experienced in performance tuning and have a large environment (more than 100 active jobs), see "Specialized Tuning" on page 13-15.

The final section of this chapter discusses the dynamic tuning capabilities of the AS/400 system. When applying changes to your system tuning, you may want to consider using techniques that still allow you to operate dynamic tuning in the new environment. There are many options for tuning your system. The concepts presented here give you some general guidelines, not all the answers. Each system environment is unique, requiring you to observe performance and make adjustments that are best for your environment.

Automatic System Tuning

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

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

Storage Pools and Activity Levels

Use the QPFRADJ system value to control automatic tuning by using storage pools and activity levels. There are two forms of automatic performance adjustments using storage pools and activity levels: initial program load (IPL) adjustments and dynamic adjustments. If performance tuning is new to you, you should set up the system to adjust at IPL and dynamically.

Adjusting Performance at Initial Program Load: If you want the system to do only initial tuning for you, set system value QPFRADJ to 1. Each time you do an IPL, the system examines the machine configuration information and makes performance adjustments to achieve efficient use of system resources. No further performance adjustments occur until you do an IPL to the system again, select dynamic performance adjustments, or issue CL commands that change the performance values. For a description of what happens during IPL performance adjustments, see "Initial Program Load Performance Adjustments" on page 13-18.

Adjusting Performance at IPL and Dynamically: If you want the system to do initial tuning for you and to dynamically make performance adjustments, set system value QPFRADJ to 2. When the system is started and periodically thereafter, the system examines the machine configuration, the jobs running on the system, storage requirements, and so on, and makes performance adjustments. Performance values settings change periodically to improve resource use on the system. For more details on the dynamic performance adjustments, see "Dynamic Performance Adjustments" on page 13-18.

Adjusting Performance Dynamically: If you want the system to dynamically make performance adjustments and **not** do initial tuning during an IPL, set the system value QPFRADJ to 3. The performance values will not be reset at IPL to the initial values.

Storage Pool Paging

To control performance using storage pool paging, you can:

- Use the Change Shared Pool (CHGSHRPOOL) command
- Use the Change Pool Attributes (QUSCHGPA) API
- Use the Change Pool Tuning Information (QWCCHGTN) API

Using the CHGSHRPOOL Command:

The paging parameter has two settings for shared storage pools: fixed paging (*FIXED) and dynamic paging (*CALC). If performance tuning is new to you, you should set this attribute to *CALC.

FIXED: The system limits the amount of memory used by jobs that are running in the storage pool. The system transfers data from auxiliary storage and frequently writes changed data back to auxiliary storage.

CALC: The system automatically determines the best approach for handling data in the storage pool. When many jobs are running in a small storage pool, the system limits the amount of memory that each job uses. When the storage pool has enough memory for the number of jobs that are running, the system determines how much data to bring into the storage pool.

For more details on the dynamic paging performance adjustments, see "Dynamic Storage Pool Paging" on page 13-20. See the *System Programmer's Interface Reference* for details on the QWCCHGTN or QUSCHGPA APIs.

Performance Components

Achieving efficient system performance requires a proper balance among system resources. Overusing any resource affects performance. You can make the best use of each resource by using proper settings for the work management parameters. For more details on the dynamic performance adjustments, see "Dynamic Performance Adjustments" on page 13-18.

Job States

Jobs running on the system can be in any of the following states:

- Active
- Wait
- Ineligible

An **active job**, exists in main storage and processes work requested by the application. A job in the **wait state** needs a resource that is not available. An **ineligible job** has work to do, but the system is unable to accept more work at that time.

Wait States: When a job is waiting for a resource, it may wait in main storage or it may be removed from main storage until the resource is available. The terms short wait, short wait extended, and long wait are used to describe a job waiting for system resource.

A job in **short wait** holds an available activity level while waiting for an activity to occur. A job can remain in short wait for a maximum of 2 seconds. Some typical causes of short waits are:

- Sending a WRITE instruction to a display when *NO is specified on the Defer Write (DFRWRT) parameter
- Sending break messages to work stations
- Specifying *YES on the Restore Display (RSTDSP) parameter on display files

When using remote lines, avoid short waits because they cause the wait time in main storage to be much longer for a job than if the job was waiting for resources at a local work station.

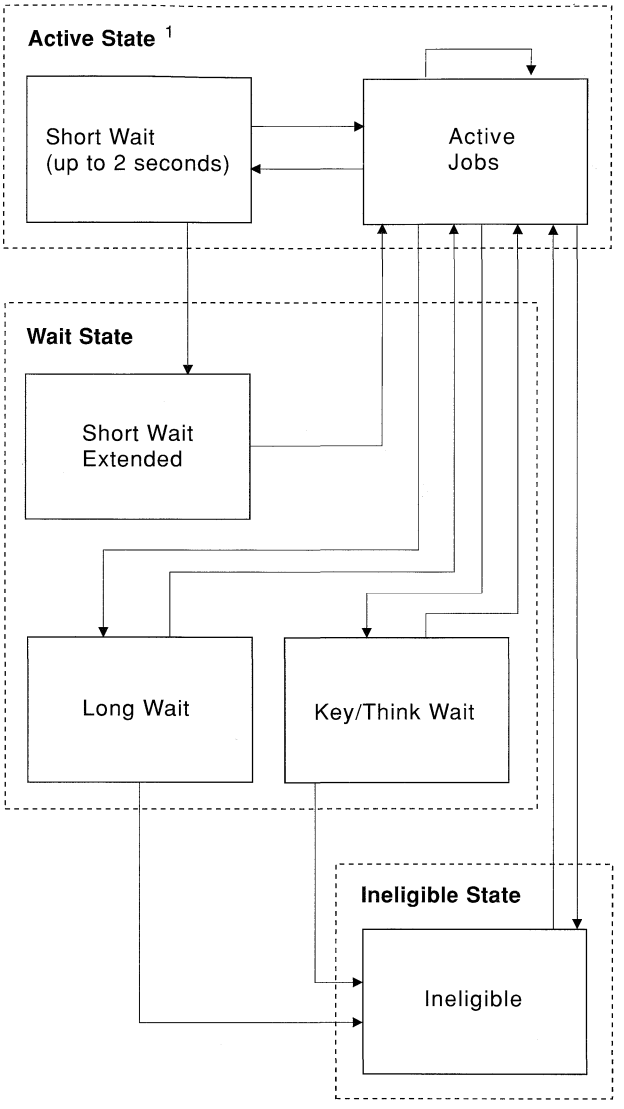
A job is in **short wait extended** if it has been in short wait for the maximum 2 seconds. After it has been in short wait for 2 seconds and the activity has not occurred, the system cancels the short wait, takes the job out of the activity level, and puts the job into a long wait. In the performance reports, this job state transition is called a short wait extended.

A job that immediately leaves the activity level is in what is called **long wait**. The long wait and the short wait extended differ from a short wait in that the job leaves the activity level during a long wait or short wait extended, but not during a short wait.

A specialized form of long wait, called **key/think wait**, occurs outside the activity level when a job completes a work assignment and returns to request more work. This is a user-specified time period giving the user time to decide what data should be entered and to type this data. When the job receives a new assignment or runs out of time (times out), it attempts to run again. If no activity level space is available, the job becomes ineligible. Other examples of long waits are:

- Record lock conflicts
- Distributed Data Management data requests

The transition from one job state to another is shown in Figure 13-1.



[1] A job in the active state occupies an activity level; jobs in the wait or ineligible state do not.

RV2W789-0

Figure 13-1. Job State Transitions

Activity Levels and Ineligible Queue

For interactive environments, typically there are more jobs running than there is space for them to run. When a job attempts to run, there must be space for the job in main storage. To restrict the number of jobs in main storage at one time, the activity level is specified for each pool in the system. Before a job can become active, an activity level must be available.

If an activity level is available, the job becomes active and begins processing in main storage. If no activity level is available, the job becomes ineli-

gible. When a job becomes ineligible, it is placed in the ineligible queue until an activity level is available.

If the job enters the long wait state by other than a lock conflict, it is placed behind all other jobs of equal priority already on the ineligible queue. This is called a **first-in, first-out priority queue**.

However, if a job becomes ineligible after a short wait extended or a long wait caused by a lock conflict, it is placed in front of jobs of equal priority already on the ineligible queue. The most common reasons for this change to normal queue placement are:

- The job entered a long wait as a result of a lock conflict because it was active (referring to objects in main storage) before the conflict occurred. If the wait was short (and many are), you may be able to get the job back into an activity level before all of the objects the job was using are removed from main storage.
- When the job has been granted the lock, it leaves the wait state. If other jobs on the ineligible queue are to use the same object, they must wait until the object is once again available. Therefore, you want jobs holding locks on objects to use them and make them available for other jobs to use. To accomplish this, the job moves ahead of any potential requesters.

By correctly managing the ineligible queue, the system may avoid unnecessary job transitions and disk operations. As a result, throughput and response time are more consistent.

System Objects

Each job running in the system is assigned to a storage pool. When a job is active, it resides in its assigned storage pool. Active jobs refer to many different system objects. When jobs use these objects, they must be in main storage. If they are not in main storage, they must be read into main storage from their locations on disk (auxiliary) storage. Some of the objects used by jobs are:

- Data areas
- File override information
- Device files
- Application codes
- System codes
- Open file information
- Queues
- Logical files
- Subfile work areas
- Physical files
- Program variables

Process Access Groups: A **process access group** (PAG) is a group of job-related objects that can be paged in and out of storage in a single operation when a job (process) enters or leaves a long wait. Although many different object types are found in the PAG, they fall into two main categories: objects that are shared by jobs and objects that are unique to a specific job. When an object is shared, only one copy of the object exists. For example, application code used by 20 jobs resides in main storage in only one place but is used by all the jobs. However, the variables and data in the application do not have the same values for all jobs using the application. These portions of the application and other unique objects are packaged as an object called a process access group (PAG).

When a job enters an activity level, the PAG is automatically transferred from auxiliary storage to main storage. After the PAG has been written to auxiliary storage, the main storage space is available for other jobs. This activity is similar to swapping. Whenever a job is active, the pages of the PAG that are actually used must be in main storage.

PURGE Parameter

PURGE is a work management tuning parameter. To get a job's PAG into main storage, the system refers to the value specified on the PURGE parameter in the job class, which is resolved when the job first enters the system. The value for the PURGE parameter is either *YES or *NO.

PURGE (*YES)

The following list describes the characteristics of the PURGE parameter when *YES is specified:

- Adapts to work load and storage size and operates as
  - *YES in limited storage
  - *NO in adequate storage

- In systems with a small amount of main storage, selecting *YES instead of *NO results in better performance.

Specifying *YES relieves you of deciding which value to use and is the proper choice for most environments. Figure 13-2 illustrates the manner of transfer when the job class is set to PURGE (*YES).

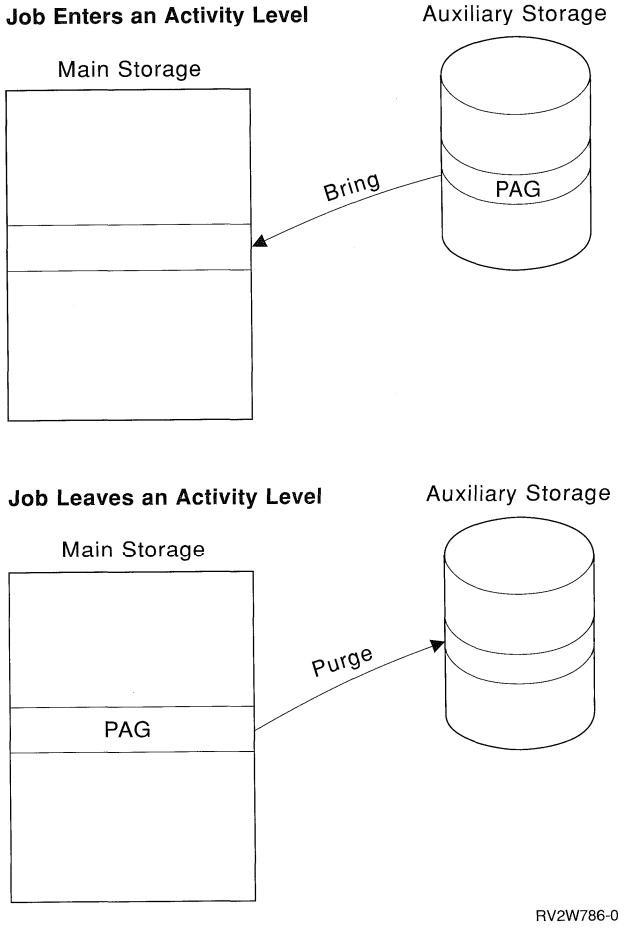


*Figure 13-2. PURGE Parameter When Specifying *YES*

If your system has enough main storage for the PAGs that are used, the system evaluates the efficiency of automatically reading and writing the PAG. If, when a job enters an activity level, the system determines that the job's PAG is already in main storage, the system does not try to read the entire PAG. Similarly, when the job leaves an activity level, the job's PAG is not automatically written to auxiliary storage. This process is called dynamic PURGE and is specified by PURGE (*YES).

PURGE (*NO)

When the job class is set to PURGE (*NO), the system does not read any portion of the PAG until the job requires a part of the PAG that is not in main storage. When this occurs, a small portion of the PAG, starting with the requested data, is read into main storage. When the job leaves an activity level, none of the job's PAG is written to auxiliary storage. The job's PAG remains in main storage until some job currently in an activity level requires more main storage. The system assigns main storage to the job that is currently active. At this point, a small portion of the inactive job's PAG is written to auxiliary storage.

The following list describes the characteristics of the PURGE parameter when *NO is specified:

- May use fewer nondatabase READ operations per transaction
- May cause fewer WRITE operations per transaction
- May transfer fewer pages
- May reduce disk use
- May use less processing unit time per transaction
- Requires more main storage

Figure 13-3 on page 13-6 shows the manner of transfer when the job class indicates PURGE (*NO).

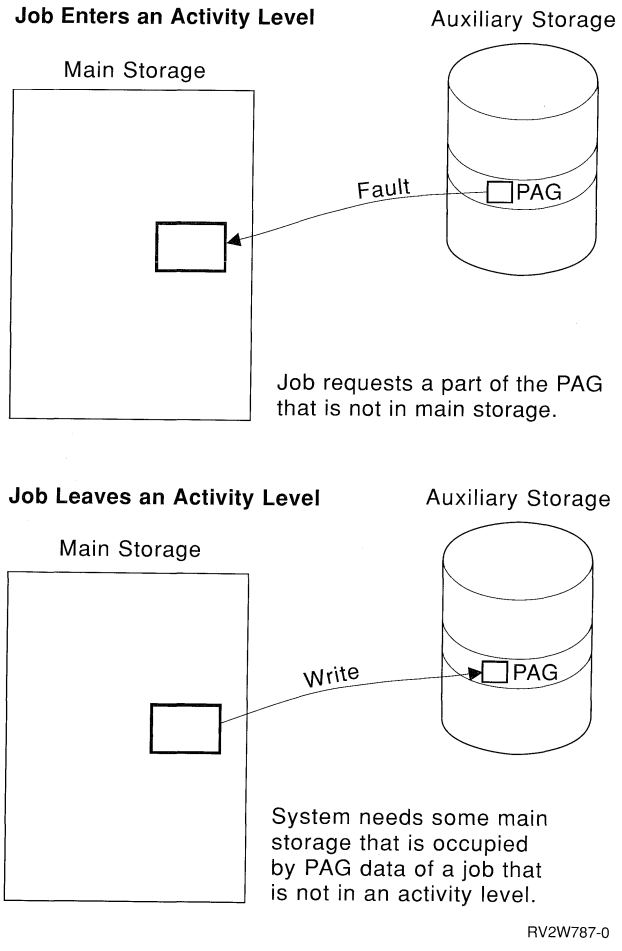


*Figure 13-3. PURGE Parameter When Specifying *NO*

Time Slice

Time slice is another of the work management tuning parameters. The time slice value is specified in the job's class. The value represents the amount of processing unit time a job is allowed to use while processing a transaction. It does not represent the elapsed time of a transaction.

If a job fails to complete a transaction in the specified time slice, one of the following occurs:

- If no jobs of equal or higher priority are on the ineligible queue, the job is given another time slice, remains in main storage, and continues the transaction.
- If jobs of equal or higher priority are on the ineligible queue, the job currently running is removed from main storage and placed on the ineligible queue. A job from the ineligible queue is moved into main storage and processing continues.
- If the job is interactive and a time slice end pool is specified for the job, the job is moved to the time slice end pool.

Long-Running Interactive Transactions

As explained in "Time Slice," a job is allocated a time slice (an amount of processing time) when it begins to process a transaction. The time slice is used to prevent processing unit-intensive transactions from using all the resources of the pool in which the transaction is running. When a job fails to complete a transaction within its assigned time slice, one of the following may occur:

- The job moves to another pool.
- The transaction run is temporarily suspended by the system.

If you specify *BASE in the system value QTSEPOOL, the system attempts to reduce the impact of a long-running transaction on other users in the pool. To do this, the system runs the remainder of the transaction in the *BASE pool rather than allowing it to complete in the interactive pool.

At the completion of the long-running transaction, the system returns the job to the interactive pool. This action assumes that batch is running in *BASE and that the time slice for the interactive jobs is exceeded by only a small percentage of transactions. Usually the transactions that exceed the default time slice value of two seconds for an interactive job are transactions that are characteristic of batch-type activity.

In general, these actions have a positive impact on system performance and *BASE should be specified for this system value. However, the following situations may cause a negative impact on system performance:

- *BASE is very small.

  If the pool is too small, there is not enough storage to contain the work being moved to the pool. When this occurs, the system begins to perform a large number of disk operations. As a result, jobs are unable to perform productive disk requests and performance is poor. If this situation occurs in your environment, add storage to the *BASE pool.

- The activity level in *BASE is not set properly.

  If the activity level is too large, either add storage to the *BASE pool or reduce the activity level. If the activity level is too small, increase the activity level and increase the main storage in *BASE. Jobs running in *BASE should have about 500KB (KB equals 1024 bytes) per activity level to perform efficiently.

By properly sizing *BASE and choosing an appropriate activity level, moving long-running transactions from the interactive pool to *BASE should provide better system performance. If system performance is not better after several tries, set QTSEPOOL to *NONE and reset the system pool sizes and activity levels to their original values.

Performance Commands

Three system commands are available to help you observe the performance of your system:

- Work with System Status (WRKSYSSTS)
- Work with Disk Status (WRKDSKSTS)
- Work with Active Jobs (WRKACTJOB)

Also, you can use the Performance Tools/400 licensed program to help analyze your performance. 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. This section discusses only the system commands. To observe the system performance, complete the following steps:

1. Use the WRKSYSSTS, WRKDSKSTS, or WRKACTJOB command.
2. Allow the system to collect data for a minimum of 5 minutes.
3. Press F5 (Refresh) to refresh the display and present the performance data.
4. Tune your system based on the new data.

Then press F10 (Restart) to restart the elapsed time counter.

Working with System Status

The Work with System Status display shows the current status of the system. When tuning the system, make sure that the machine pool is treated separately from the other pools. Type the WRKSYSSTS command on the command line and press the Enter key. The Work with System Status display appears:

```
                          Work with System Status                 SYS01
                                                          02/23/90  09:52:11
% CPU used . . . . . :       35.6    Auxiliary storage:
Elapsed time . . . . :   00:01:32      System ASP . . . . . :     1803 M
Jobs in system . . . :         96      % system ASP used: . :    64.3508
% addresses used:                      Total  . . . . . . . :     1803 M
  Permanent  . . . . :      2.805      Current unprotect used :    285 M
  Temporary  . . . . :      5.906      Maximum unprotect  . . :    307 M

Type pool size and activity level changes, press Enter.

System    Pool    Reserved   Max    -----DB-----  ---Non-DB---
 Pool   Size (K)  Size (K)  Active  Fault  Pages  Fault  Pages
   1       9639      4517    +++      .0     .0     .3     .4
   2       4000         0      6      .0    5.4    1.8    4.0
   3      35163         0     50      .0     .2    1.1    6.5
   4        350         0      5      .0     .0     .0     .0

                                                                  Bottom
Command
===>
F3=Exit    F4=Prompt           F5=Refresh    F9=Retrieve      F10=Restart
F11=Display transition data    F12=Cancel    F24=More keys
```

Figure 13-4 shows the nondatabase fault rate. Since the machine pool contains objects used system-wide, page faulting in the machine pool affects all jobs on the system. Therefore, it is desirable to maintain a low page fault rate in this pool. The guidelines you should apply are listed in Figure 13-4.

Figure 13-4. Nondatabase Paging Faults

Main Storage Size	Good	Acceptable	Poor
Less than or equal to 12MB	<2	2 – 5	>5
More than 12MB	<1	1 – 3	>3

The only way to affect paging in the machine pool is to adjust the size of the pool (values are represented in the second column of the Work with System Status display).

As the Work with System Status display shows, all other pools require attention to both database and nondatabase page faulting rates and the job transitions. The guidelines for the faulting rates in these pools are based on the sum of the database and nondatabase faults per second. In addition, you can adjust the processor speed and the PURGE attribute.

Figure 13-5 on page 13-8 shows the general guidelines for each pool.

Figure 13-5. Sum of Database and Nondatabase Faulting Levels per Pool

	PURGE(*YES)			PURGE(*NO)		
Model	**Good**	**Acceptable**	**Poor**	**Good**	**Acceptable**	**Poor**
B10 B20 C04 C06 C10 C20 D02 D04 D06 D10 E02 E04 F02	<10	10–15	>15	<15	15–20	>20
B30 B35 B40 B45 C25 D20 D35 E06 E10 F04	<15	15–20	>20	<20	20–25	>25
B50 B60 B70 D25 D45 D50 D60 E20 E25 E35 E40 E50 E60 F06 F10 F20 F25 F35 F45 F50	<20	20–30	>30	<25	25–30	>30
D70 D80 E70 E80 F60 F70	<25	25–35	>35	<30	30–40	>40
E90 E95 F80	<25	25–35	>35	<35	35–45	>45
F90 F95	<30	30–40	>40	<40	40–50	>50

In addition, you should observe the total number of faults in all pools. Figure 13-6 on page 13-9 shows the general guideline for the total of the database and nondatabase faults per second in all pools.

Figure 13-6. Sum of Database and Nondatabase Paging Faults in All Pools

Model	Good	Acceptable	Poor
B10 B20 C04 C06 C10 C20 D04 D06 D10 E02 E04 F02	<20	20–25	>25
B30 B35 B40 B45 C25 D20 D35 E06 E10 F04	<25	25–35	>35
B50 B60 B70 D25 D45 D50 D60 E20 E25 E35 E40 E50 E60 F06 F10 F20 F25 F35 F45 F50	<30	30–45	>45
D70 D80 E70 E80 F60 F70	<40	40–50	>50
E90 E95 F80	<50	50–65	>65
F90 F95	<60	60–75	>75

If you want information about the transition data, press F11 to obtain the following display:

```
                          Work with System Status                 SYS01
                                                        02/23/90  09:52:11
% CPU used . . . . . . . :       35.6    Auxiliary storage:
Elapsed time . . . . . . :   00:01:32      System ASP . . . . . :     1803 M
Jobs in system . . . . . :         96      % system ASP used. . :    64.3508
% addresses used:                          Total . . . . . . . :     1803 M
  Permanent  . . . . . . :      2.805      Current unprotect used :   285 M
  Temporary  . . . . . . :      5.906      Maximum unprotect  . . :   307 M

Type pool size and activity level changes, press Enter.

System    Pool    Reserved   Max    Active->  Wait->  Active->
 Pool   Size (K)  Size (K)  Active    Wait     Inel     Inel
   1       9639      4517    +++       2.6       .0       .0
   2       4000         0      6      77.5       .0       .0
   3      35163         0     58      18.2       .0       .0
   4        350         0      5        .0       .0       .0

                                                                   Bottom
Command
===>
F3=Exit   F4=Prompt     F5=Refresh   F9=Retrieve   F10=Restart
F11=Display pool data   F12=Cancel   F14=Work with Subsystems   F24=More keys
```

When observing job transitions:

- Wait-to-ineligible transitions need not be 0 all the time. In heavy-use periods, it may be advisable to cause jobs to become ineligible to avoid excessive page fault rates.

  To determine the proper number of wait-to-ineligible transitions, divide the number of wait-to-ineligible transitions by the active-to-wait transitions. Compare your results to the following list.

  Good <.1
  Acceptable .1 – .25
  Poor >.25

- It is usually advisable to complete a transaction during a single time slice. This reduces the number of times the job enters and leaves main storage. Therefore, the active-to-ineligible transitions should be 0. However, long running transactions should not occupy activity levels for the entire transaction. You should establish a time slice value that allows 90% of the transactions in your environment to finish in a single time slice (for example, 3 times the average CPU per transaction).

When using this display, remember that page fault rates are much more important than the job transition values. If you correctly tune the page fault rates, the transition rates usually fall within the guidelines.

You can increase or decrease the pool size or the activity level to get the desired values for pools 2 through 16. The mechanics of these actions are discussed under "Basic Tuning" on page 13-11.

Working with Disk Status

The WRKDSKSTS command shows you your system's disk activity and helps you determine the performance capabilities of your system's disks. The *CL Reference* manual contains a description of the WRKDSKSTS command and formatting information. Type the WRKDSKSTS command on the command line and press the Enter key. The Work with Disk Status display appears:

```
                          Work with Disk Status                   SYS01
                                                        02/23/90  11:04:19
Elapsed time:   00:01:55

             Size    %     I/O   Request   Read  Write   Read  Write    %
Unit  Type    (M)  Used    Rqs  Size (K)   Rqs   Rqs    (K)   (K)   Busy
  1   9332    200  98.9    .4      1.2      .0    .3    1.9   1.0      1
  2   9332    200  60.1    .6      1.0      .4    .2    1.0   1.0      2
  3   9332    200  61.8    .5      1.6      .2    .2    1.4   1.7      2
  4   9332    200  61.5    .6       .9      .3    .3     .9    .8      2
  5   9332    200  61.7    .7      1.9      .3    .4    2.5   1.4      2
  6   9332    200  60.5    .3      1.2      .1    .2    2.0    .7      1
  7   9332    200  63.2    .9      1.2      .3    .5    1.8    .9      3
  8   9332    200  56.3    .8      5.8      .4    .4    3.0   9.1      3
  9   9332    200  56.1    .6      4.2      .2    .3    1.3   6.4      2




                                                                   Bottom
Command
===>
F3=Exit   F5=Refresh   F12=Cancel   F24=More keys
```

Note: This display may vary up to 20% from actual values.

Before observing disk status, tune your system according to the paging guidelines described in the topic "Working with System Status" on page 13-7. When viewing the Work with Disk Status display, observe the percent busy data. Each unit (actuator) should be less than 40% busy. If each unit is between 40% and 60% busy, you may experience variable response times. If each unit is more than 60% busy, you do not have enough actuators to provide good performance. The **actuator** is the device within an auxiliary storage device that moves the read and write heads. If you have a well-tuned system with actuators that exceed 40% busy, you should increase the number of disk actuators.

It is possible to experience inadequate performance even if only one actuator exceeds the 40% busy guideline. This may be caused by the placement of frequently used data on a single actuator. If this occurs on your system, use the Performance Tools/400 licensed program disk report to determine which data is causing the heavy use. You can save, delete, or restore some objects to improve performance.

An actuator may exceed the 40% guideline for a short period of time. This condition may be caused by a batch job that is accessing data. If the data is not concentrated on a particular actuator, the high level of use should migrate from actuator to actuator while the batch job is running. Also, an actuator in an auxiliary storage pool (ASP) may be used heavily. But typically this is not considered to exceed the guidelines. If you observe this activity, do not change the disk configuration.

Working with Active Jobs

The WRKACTJOB command measures system performance. The following examples show the WRKACTJOB displays.

To view the Work with Active Jobs display, type WRKACTJOB on any command line, and press the Enter key.

```
                              Work with Active Jobs                  SYS01
                                                          02/23/90  11:04:44
CPU %:     .0     Elapsed time:   00:00:00     Active jobs:   67

Type options, press Enter.
  2=Change   3=Hold   4=End       5=Work with      6=Release
  7=Display message   8=Work with spooled files    13=Disconnect...

Opt  Subsystem/Job  User        Type  CPU %  Function        Status
     QBASE          QSYS        SBS      .0                 DEQW
       USER1        USER1       EVK      .0                 ICFW
       USER1        USER1       EVK      .0  *  -PASSTHRU    EVTW
       USER1        USER1       EVK      .0  *  -PASSTHRU    EVTW
       USER1        USER1       EVK      .0  *  -PASSTHRU    EVTW
       USER2        USER2       EVK      .0  *  -PASSTHRU    EVTW
       USER2        USER2       EVK      .0  *  -PASSTHRU    EVTW
       USER3        USER3       EVK      .0  *  -PASSTHRU    EVTW
       DSP020000    PGRENGS     INT      .0  MNU-PROGRAM     DSPW
                                                                More...
Parameters or command
===>
F3=Exit      F5=Refresh   F10=Restart statistics   F11=Display elapsed data
F12=Cancel   F23=More options   F24=More keys
```

To display information about elapsed data, press F11 to obtain the following display:

```
                              Work with Active Jobs                  SYS01
                                                          02/23/90  11:04:44
CPU %:     .0     Elapsed time:   00:00:00     Active jobs:   6/

Type options, press Enter.
  2=Change   3=Hold   4=End      5=Work with      6=Release
  7=Display message   8=Work with spooled files  13=Disconnect...
                                                 --------Elapsed---------
Opt  Subsystem/Job  Type  Pool  Pty      CPU  Int    Rsp  AuxIO  CPU %
     QBASE          SBS      2    0    813.8                 0     .0
       USER1        EVK      2   20      2.1                 0     .0
       USER1        EVK      2   50       .4                 0     .0
       USER1        EVK      2   50       .2                 0     .0
       USER1        EVK      2   50       .2                 0     .0
       USER2        EVK      2   50       .1                 0     .0
       USER3        EVK      2   50       .2                 0     .0
       USER4        EVK      2   50       .2                 0     .0
       DSP020000    INT      3   20      15.7    0     .0     0     .0
                                                                  More...
Parameters or command
===>
F3=Exit      F5=Refresh        F10=Restart statistics   F11=Display status
F12=Cancel   F23=More options   F24=More keys
```

Use both the WRKSYSSTS and the WRKACTJOB commands when attempting to observe your system's performance. With each observation period, you should examine and evaluate the measures of system performance against the goals you have set. Some of the typical measures include:

- Interactive throughput and response time, available from the WRKACTJOB display.
- Batch throughput. Observe the AuxIO and CPU% values for active batch jobs.
- Spool throughput. Observe the AuxIO and CPU% values for active writers.

Each time you make tuning adjustments, you should measure and compare all of your key performance measures. Make and evaluate adjustments one at a time.

Note: While using the WRKACTJOB command, if you position the cursor on a column and press F16, the items displayed are sorted according to the entries in that column.

Basic Tuning

This section describes some of the steps you can take to initially configure the system pool sizes and activity levels to tune your system efficiently. The following topics are discussed:

- Initial machine pool size
- Choosing pool configuration
- Adjustments to pool sizes and activity levels

Initial Machine Pool Size

Maintaining low page fault rates in the machine pool helps the system perform better. The following formula is intended only as a guideline to be used when setting the initial machine pool size:

```
S = M + J + L + F
```

where:

S Initial machine pool size
M Minimum machine pool size (see Figure 13-7)
J Job space (see Figure 13-8)
L Communications space (see Figure 13-9 on page 13-12)
F Functional space (see Figure 13-10 on page 13-12)

Minimum Machine Storage Size: To find the value for the minimum machine pool size, locate your main storage size in Figure 13-7 and use the corresponding value.

Figure 13-7. Minimum Machine Pool Size

Main Storage Size (MB)	Minimum Machine Pool Size (KB)
4	1175
8	1625
12	2050
16	2400
20	2750
24	3050
28	3350
32	3650
36	3950
40	4250
48	4800
64	6200
72	6900
80	7600
96	9000
112	10300

Figure 13-7. Minimum Machine Pool Size

Main Storage Size (MB)	Minimum Machine Pool Size (KB)
128	11600
144	12800
160	14000
192	16200
208	17300
224	18400
240	19500
256	20600
272	21700
288	22800
304	23900
320	25000
336	26100
352	27200
384	29400
416	31600
448	33800
480	36000
512	38200
576	42600
640	47000
704	51400
768	55800
832	60200
896	64600
960	69000
1024	73400
1088	77800
1152	82200
1216	86600
1280	91000

Job Space: Job space is set aside in the machine pool for each active job in the system. Use Figure 13-8 to get a value for this part of the system. However, a more accurate value is derived if you:

- Estimate the maximum number of jobs (including group jobs) that are active at the same time.
- Add the value you estimate to the value obtained from Figure 13-7, because each job requires 1KB in the machine pool.

Figure 13-8 (Page 1 of 2). Job Space

Main Storage Size (MB)	Job Space
≤12	20—80
16—28	55—130
32—48	95—240

Figure 13-8 (Page 2 of 2). Job Space

Main Storage Size (MB)	Job Space
56—192	130—680
208—272	550—855
288—384	890—1070
416—512	1130—1310
512—768	1310—2000
768—1024	2000—2650
1024—1280	2650—3300

Communications Space: For each line, protocol, line type, and controller, the machine pool must include some additional main storage. Figure 13-9 is given as a simple guide to the additional storage required. These values are based on estimates of the numbers and types of communications lines that may appear for each model and main storage size. However, you can do a better job because you know the communications configuration of the system. To estimate additional main storage:

1. Add 125KB for each line.
2. Add 100KB for each protocol (BSC, SDLC, asynchronous, and so on).
3. Add 25KB for each remote work station controller.
4. If you have a 2617 Ethernet or a 2619 token ring card, add 1.5MB for each card. You can ignore adding 125KB for each line associated with each card.

Figure 13-9. Communications Space

Main Storage Size (MB)	Communications Space
≤12	250—525
16—28	500—1000
32—48	1050—1750
56—192	1525—5100
208—272	4200—6550
288—384	6800—8100
416—512	8500—9700
512—768	9700—15 000
768—1024	15 000—20 000
1024—1280	20 000—25 000

Functional Space: Finally, certain system functions, when active, require additional space in the machine pool. If you plan to use any of the functions shown in Figure 13-10, you should add the appropriate amount of main storage to the machine pool.

Figure 13-10. Functional Space

Function	Addition to Machine Pool
3270 emulation or remote attachment	50KB
Checksum	5% of main storage size
Save or restore	68KB[1]
Double-byte character set	50KB
X.25	48KB
Token-ring local area network[2]	250KB
OfficeVision/400	600KB

Notes:

1 For each Save or Restore operation occurring at the same time.

2 See "Communications Space" for specific allocation amounts.

Choosing Your Pool Configuration

After you have set the machine pool size, you need to decide what to do with the remaining storage. Your choices include:

- To create no additional pools
- To create a separate pool for interactive jobs, for example the *INTERACT shared pool in the QINTER subsystem
- To create a separate pool for spool jobs, for example the *SPOOL shared pool in the QSPL subsystem

If you choose to create separate pools for QINTER and QSPL, you can add the values from Figure 13-11 and Figure 13-12 to determine the initial values for the size and activity levels for the spool pools. Then use Figure 13-13 on page 13-13 to calculate the *BASE pool sizes and activity levels. Batch jobs (including System/36 environment-started batch jobs) run in the *BASE pool. The remaining storage is the QINTER pool size. To calculate an activity level for the QINTER pool, divide the pool size by the value determined from Figure 13-14 on page 13-13.

Figure 13-11 on page 13-13 shows the QSPL pool sizes and activity levels for advanced function printers.

Figure 13-11. QSPL Pool Sizes and Activity Levels for Advanced Function Printers

Number of Writers	Size (KB)	Activity Level
1	1500	1
2	1700	2
3	1900	3
4	2100	4
>4	2300	5

Figure 13-12 on page 13-13 shows the QSPL pool sizes and activity levels for non-advanced function printers.

Figure 13-12. QSPL Pool Sizes and Activity Levels for Non-Advanced Function Printers

Number of Writers	Size (KB)	Activity Level
1	80	1
2	160	2
3	225	3
4	290	4
>4	350	5

The activity level reflects an estimate of the maximum number of batch jobs (including called programs) that can be active at the same time as the interactive work. Figure 13-13 assumes that interactive work is running at the same time as batch jobs.

Figure 13-13. Pool Size (KB)/ Activity Level

Main Storage Size (MB)	Pool Size (KB)/ Activity Level
≤12	500/2—1250/3
16—28	1500/4—2350/5
32—48	2700/5—4000/6
56—192	4625/5—14 000/7
208—272	15 000/8—18 600/9
288—384	19 500/9—23 100/10
416—512	24 000/15—27 900/15
512—768	27 900/15—37 500/20
768—1024	37 500/20—47 100/20
1024—1280	47 100/20—56 000/25

Figure 13-14 shows the activity level factor for the QINTER pool:

Figure 13-14. QINTER Activity Level Factor

Main Storage Size (MB)	Activity Level Factor (KB)
≤ 12	450KB
16 — 28	900KB
32 — 48	1600KB
64 — 192	2500KB
208 — 272	3000KB
288 — 384	3500KB
416 — 512	4000KB
512—768	4500KB
768—1024	5000KB
1024—1280	5500KB

Adjustments to Pool Sizes and Activity Levels

Once you have set initial pool sizes and activity levels, you should begin to observe system performance. When you are observing performance:

- Use both WRKSYSSTS and WRKACTJOB commands.
- Observe at intervals of 5 minutes.
- Observe when the system is handling *normal* work loads (for example, observations during the noon hour may not give meaningful data).
- Apply the page fault rate guidelines shown in Figure 13-5 on page 13-8.

If your observations indicate page fault rates outside the stated guidelines, you should determine the cause of the high page fault rate. Do not make adjustments to tuning if some abnormal condition is causing the high page fault rate. Interactive program compiles, communications error recovery procedures (ERP), open query file (OPNQRYF), application errors, and sign-off activity are examples of irregular activities that may cause a severe hit to performance. Rather than adjusting the system tuning, you should try to remove as many irregular activities as possible from your operating environment.

If you decide that the high page fault rates in your system occur during normal system operation, adjust the tuning parameters, primarily pool size and activity levels. Tune the machine pool first. You can only reduce machine pool page faulting by increasing the size of the pool. Once you have reached the values shown as *good* in Figure 13-4

on page 13-7, Figure 13-5 on page 13-8, and Figure 13-6 on page 13-9, you can stop.

If your machine pool is experiencing very low page fault rates (<0.4), you should decrease the size of the machine pool. If the page fault rate is this low, you may be affecting work in some other pools.

After tuning the machine pool fault rate, you need to focus on the other pools in the system. Once again, you can adjust size and activity level.

Adjusting Activity Levels

Adjusting the activity level may be the most beneficial way to improve the tuning in pools 2 through 16. The possible problems that can exist in these pools are:

- Low fault rate with an unacceptable value of wait-to-ineligible transitions

  In this situation, the activity level is probably set too low. The paging rate is low because only a few jobs are allowed in the pool at the same time. As a result, the jobs must wait for an activity level before they can run. Increase the activity level by one until the paging rate in the pool and the wait-to-ineligible transitions reach an acceptable level. Be certain to measure and compare each change so you can identify the best activity level.

- A moderate fault rate with an acceptable value of wait-to-ineligible transitions

  This situation indicates that the activity level may be set correctly, but the pool size is too small. To alleviate this condition, configure more storage and, possibly, more activity levels for this pool.

- A high fault rate with either a good or an unacceptable value of wait-to-ineligible transitions

  In this situation, the activity level is probably set too high. In this case, too many jobs are allowed into the pool at the same time. Instead of doing productive work in main storage, the jobs must read information to complete their transactions many times. Demand for space in the pool is so high that jobs are unable to use their pages before another job has overlaid their information. This condition is called **thrashing**. To reduce thrashing, reduce the activity level by one until paging returns to an acceptable rate. Once again, measure and observe performance after each change.

As you adjust the activity level, you may find that you are cycling through the situations described above. Using the data for the WRKSYSSTS and WRKACTJOB commands, select the tuning values that have provided the best performance. If performance is still inadequate, use the Start BEST/1* (STRBEST) command of the AS/400 Performance Tools to evaluate system upgrades to main storage, disk, and the processing unit.

When applying these tuning techniques, change the value of QPFRADJ to `0`. Otherwise, the system changes the settings of the performance values that you have set.

Adjusting Pool Sizes

After your activity level is at its optimum setting, examine the sizes of the pools in your system. If you detect a pool with a low fault rate and a low wait-to-ineligible count, the pool probably has too much storage. Reduce the size of storage-rich pools and add size to the pools with less storage. Reduce size in decrements of 10%. Once again, measure each change.

A change in pool size takes effect immediately. See "Changing the Size of a Storage Pool" on page 3-18 for information on how to change pool sizes.

Reviewing Performance

Once you have set good tuning values, you should periodically review them to ensure your system continues to do well. 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 may achieve your objectives. If, after a reasonable effort, you are still unable to meet your objec-

tives, you should determine whether your objectives are realistic for the type of work you are doing.

Specialized Tuning

After you have had some time to observe your system's performance, you may want to consider more specialized tuning for your environment. Some considerations are:

- Using PURGE(*NO) for interactive jobs
- Separating batch work from *BASE
- Using multiple pools for interactive jobs
- Using multiple pools for batch jobs

Because the system does authority checking on an object each time it is accessed, the best performance can be achieved by using the public authority for the object and granting no private authorities.

Specifying PURGE(*NO) for Interactive Jobs

The characteristics of interactive jobs are different when PURGE(*NO) is specified on the job class rather than PURGE(*YES). In environments with sufficient main storage, you may want to specify PURGE(*NO).

Environments that will benefit from running PURGE(*NO) are environments with:

- A lot of main storage
- A high volume of transactions

You can multiply the number of jobs running in the pool by 350KB to determine the size a pool should be if PURGE(*NO) is specified. If the result of your multiplication is approximately equal to your pool size, set the PURGE attribute to *NO. Use Figure 13-14 on page 13-13 to calculate the initial activity level of the pool.

Separate Batch Work from *BASE

To this point, batch jobs have shared *BASE storage with system jobs (for example, SCPF, QSYSARB, and subsystem monitors) and system transients (for example, file OPEN and CLOSE). As a result, batch jobs compete for space in the *BASE pool. Batch performance may improve if the jobs are moved to a separate pool. Calculate the size of the pool for each job running at the same time. Use the sizes shown in Figure 13-16 on page 13-16 as an initial estimate for each job. Set the activity level to be equal to the number of jobs running at the same time. For example, a pool will have 3 batch jobs running. One job is a compiler, one is a short-running production job, and the third is a long-running production job. In this case the pool size would be 3250KB (2000KB + 500KB + 750KB) and the activity level would be 3. If batch is not active, the storage allocated to a private batch pool is not used. This does not make efficient use of main storage. However, if you use a shared pool and choose one of the dynamic tuning options (QPFRADJ set to '2' or '3'), storage not being used by batch jobs will be allocated to pools with active jobs. Figure 13-15 shows the guidelines for the sum of database and nondatabase faults in *BASE after the batch jobs have been removed from *BASE.

*Figure 13-15. Database and Nondatabase Paging Faults in *BASE for All Models with Batch Jobs No Longer in *BASE*

Model	Good	Acceptable	Poor
B10, B20, C04, C06, C10, C20, D04, D06, D10, E02, E04, F02	< 5	5–7	> 7
B30, B35, B40, B45, C25, D20, D35, E06, E10, F04	< 7	7–10	>10
B50, B60, B70, D25, D45, D50, D60, E20, E25, E35, E40, E50, E60, F06, F10, F20, F25, F35, F45, F50	<10	10–15	>15
D70, D80, E70, E80, F60, F70	<15	15–20	>20
E90, E95, F80	<20	20–25	>25
F90, F95	<25	25–30	>30

Multiple Pools for Interactive Jobs

There are instances where distinct sets of interactive users should be isolated. Some examples are:

- Programmers
- Users performing office-type functions exclusively
- Data entry personnel

For programmers and users performing office-type functions exclusively, you are attempting to isolate users who are performing the same functions. Often, the functions performed by these users are different from the functions used by all other users. Also, some of these users may be classified as casual users, and isolating them helps protect their objects. For data entry personnel, you are isolating extremely active users to give the best possible response time for their activity.

If you are considering this approach, you need to determine how many users are to be put in each pool. Then, after setting the necessary routing entries in the subsystem description, you need to calculate a pool size, activity level, and PURGE attribute for each pool. If you are setting a separate pool for casual users:

- Set the PURGE attribute to *YES.
- Calculate the activity level by dividing by four the number of work stations to be routed to the pool.
- Multiply the activity level by 600KB to set the pool size.

Each pool should be tuned to the optimal pool size and activity level. After you have separated work into private pools, jobs can use only the storage of the pool in which they are running. If you have used shared pools, the dynamic tuner will attempt to balance storage.

Multiple Pools for Batch Jobs

Each batch job may use objects that are different from the objects used by other batch jobs in the same pool. Production batch and program compiles do not use the same objects. Long-running jobs may not perform well if sharing a pool with short-running jobs. System/36 environment evokes may disrupt other batch jobs' performance. If you have some of these situations, you may want to set up multiple batch pools.

When you separate the pools, set up a pool for each batch job type and use Figure 13-16 to find the approximate size of the jobs in the pool. Each pool should have an activity level of one and only one active job. Again, if you have used private pools and there is no work being performed in these pools, the main storage is not used by the system to support jobs in other pools. If you have used shared pools, the dynamic tuner will assign storage from an inactive pool to one that is active.

Figure 13-16. Batch Job Storage Guidelines

Batch Job Type	Initial Storage	Comments
Short-Running Production	500KB	May run in 250KB; may require as much as 750KB
Long-Running Production	750KB	May run in 500KB; runs better in 1000KB
Compiles	2000KB	May run in 1500KB; runs better in 3000KB
Reformat (Sort)	2000KB	Smaller sorts may run in 1500KB; larger sorts may use 2000 to 3000KB
Queries	2000KB	Smaller queries run in 1500KB; larger queries may use up to 4000KB
Save/Restore	2000KB	Some SAVE operations run in 1000KB; others may need 3000KB

Storage Pools and Object Selection

Note

This is an advanced performance tuning function.

If you want to reduce the disk accesses needed to perform a task, controlling individual objects and the main storage pool that contains those objects may help you.

To move an object into a specific main storage pool, use the Set Object Access (SETOBJACC) command. This command moves the object more efficiently than other functions that run on the object.

To limit the objects in main storage to only the ones you want, define a pool in a subsystem (with no jobs running in that pool). Clear the pool using the Clear Pool (CLRPOOL) command. Then use the SETOBJACC command to add an object to the pool; use the same command with the *PURGE option when you no longer need the object. The only objects that remain in the pool will be those you select using the SETOBJACC command.

In addition to bringing objects into a storage pool, the SETOBJACC command:

- Clears the object from all main storage pools before reading it into the specified pool. This allows you more control over which pools provide storage for the object.
- Transfers only those parts that existed at the time the command was issued. Additions to an object that use space that is already allocated (in a file) or cause additional disk space allocations are stored in the job's pool.
- Reports the current amount of storage used within the pool. This information is necessary to determine what objects fit in a particular storage pool.

Notes:

1. The SETOBJACC command does not affect the performance of write operations.
2. If dynamic performance tuning is active, it is recommended that you use only private pools with this command.

System/36 Environment Tuning

Within the System/36 environment, you should consider several key tuning parameters. In particular, the following may result in performance improvements:

- Set the Multiple Requester Terminal (MRT) delay value to avoid unnecessary start and end activity.
- Minimize the amount of security validation with the MRT security value.
- Use PURGE(*NO) for MRT jobs.
- Limit OCL logging.
- Route evoked jobs to the batch job pools and run them at batch priority.

When setting the MRT delay value, select a value that minimizes the number of times the MRT job starts and ends. The overhead for starting and ending on the AS/400 system is much greater than on the System/36. The MRT delay value specifies the amount of time (in seconds) that the MRT job remains active after the last work station detaches from the job. Delaying the ending of the MRT job may allow another work station to attach to the MRT while it is still active.

The MRT security parameter controls the amount of authority checking that is performed each time a work station attaches to the MRT job. The default value causes authority checking of all objects (files, libraries, and so on) used by the job for each work station that attaches to the MRT. You may set this value so that authority checking is performed only when the MRT is started. Use the default value only in cases where a user has authority to the MRT program but not to all the objects used by the program. These cases are not common.

Since the MRT job has several work stations attached, the PURGE attribute setting may influence the MRT performance. Setting the PURGE attribute to *NO for MRT jobs reduces the number of disk operations performed by the system each time a transaction is performed by any work station that is attached to the MRT.

You should also reduce OCL logging for all System/36 programs. The default value is to log all OCL commands that are run. This creates additional overhead, large job message queues, and large job logs. Set the value to log OCL commands that run by system procedures only, not application procedures. Use application procedure OCL logging only during debugging of the procedure. When an application procedure is running correctly, turn off OCL logging.

Set the priority and pool in which evoked jobs run to reduce interference with the interactive jobs. The nature of the evoked job is often that of a batch job. Therefore, any subsystem supporting evoke operations should have a routing entry that

places the evoked jobs in the *BASE pool at a running priority that is the same as batch. Use immediate initiation for evoked jobs unless you want to limit the number of evoked jobs that can be active at the same time.

By correctly setting these System/36 running parameters, you may improve performance. Each user's requirements need to be analyzed to find the best setting for each of the parameters described.

Performance Adjustments

Storage Pool Size and Activity Level Tuning

Initial Program Load Performance Adjustments:

If you would like the system to do initial tuning for you, but you do not want the system to perform dynamic tuning, set system value QPFRADJ to 1. When you perform an IPL, the system examines the machine configuration and the controlling subsystem value. If QPFRADJ is set to 1 or 2, the system uses the configuration information to set the initial pool sizes and activity levels. If the controlling subsystem is QBASE or QCTL, the system sets up separate pools for spool and interactive jobs.

The IPL performance adjustments result in changes to the following values:

- Machine pool size (QMCHPOOL system value)
- Base pool activity level (QBASACTLVL system value) if the controlling subsystem is QSYS/QBASE, QSYS/QCTL, QGPL/QBASE, or QGPL/QCTL
- Pool number 2 in subsystem QGPL/QSPL to use shared pool *SPOOL
- Pool size and activity level for shared pool *SPOOL
- Pool number 2 in subsystems QSYS/QBASE and QGPL/QBASE to using shared pool *INTERACT if controlling subsystem is QSYS/QBASE, QSYS/QCTL, QGPL/QBASE, or QGPL/QCTL
- Pool number 2 in subsystems QSYS/QINTER and QGPL/QINTER to using shared pool *INTERACT if controlling subsystem is QSYS/QBASE, QSYS/QCTL, QGPL/QBASE, or QGPL/QCTL
- Pool size and activity level for shared pool *INTERACT

These performance values are set to the values defined in "Basic Tuning" on page 13-11.

If you make any adjustments to the pool size or activity level values, you should consider setting QPFRADJ to 0. Otherwise, the values are reset at the next IPL. If you are satisfied with the QPFRADJ values but change your environment to run night batch jobs, you may wish to leave QPFRADJ set to 1 or 2.

Note: At the first IPL after the 2617 Ethernet or 2619 token-ring card is added, dynamic tuning does not adjust main pool storage for the Ethernet or token-ring networks. Main pool storage is adjusted at the second IPL to consider these cards.

Dynamic Performance Adjustments:

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 underused storage pools to pools that would benefit from more storage. This tuning also sets activity levels to balance the number of jobs in the pool with the storage allocated for the pool. The tuner uses the guidelines in "Working with System Status" on page 13-7 to adjust the system.

Dynamic tuning changes the following performance values:

- *MACHINE pool size (QMCHPOOL system value)
- *BASE pool activity level (QBASACTLVL system value)
- Pool size and activity level for shared pool *INTERACT
- Pool size and activity level for shared pool *SPOOL
- Pool sizes and activity levels for shared pools *SHRPOOL1-10

If you make any adjustments to the pool size or activity level values, you should consider setting QPFRADJ to 0. Otherwise, the values are

changed by the IPL or periodic performance adjustments.

Minimum Pool Size: When the dynamic tuning support is making changes to storage pool sizes, the dynamic tuning support ensures that the pools are not reduced beyond a minimum size. Figure 13-17 shows the minimum pool sizes for pools that are having some page faults occur.

Figure 13-17. Minimum Pool Size

Share Pool	Minimum Pool Size
*MACHINE	Minimum size determined by Vertical Licensed Internal Code (VLIC)
*BASE	Minimum size is value of QBASPOOL system value
*INTERACT	600KB
*SPOOL	80KB
*SHRPOOL 1–10	300KB

You can use the Work with System Value (WRKSYSVAL) command to change the minimum size of the *BASE pool. The minimum size used by the dynamic tuning support for pools other than the *BASE pool cannot be changed.

If no page faults occur in a pool, the dynamic tuning support may reduce the size of the *BASE, *INTERACT, *SPOOL, or *SHRPOOL1 through 10 pools to 48KB. This allows other pools to use the storage that is not being used.

Journaling Dynamic Performance Adjustments: This section describes how you can set up a journal to record the changes made to the system by the dynamic tuning support. You can track, or journal, the system so you can see which changes have been made to the system. An entry is made in the journal only if a change is made to a pool size or if pool faults occur in a pool. The changes are recorded in journal QSYS/QPFRADJ. You should know how to perform journal management operations, such as saving a journal receiver, changing journal receivers, and deleting old journal receivers. For more information about journal management, see the *Advanced Backup and Recovery Guide*.

To set up the performance adjustment journal, do the following:

1. Create a journal receiver of your choice by using the Create Journal Receiver (CRTJRNRCV) command:

   ```
   CRTJRNRCV JRNRCV(QSYS/QPFRADJ1)
   ```

   Name the journal receiver QPFRADJ1, or select a similar name you can use to create a naming convention such as QPFRADJ2 or QPFRADJ3, for future journal receivers.

   After you create the first receiver, you can use the CHGJRN JRNRCV(*GEN) command to create additional receivers and attach them to the QSYS/QPFRADJ journal automatically with the correct naming convention.

2. Create the journal QSYS/QPFRADJ by using the Create Journal (CRTJRN) command:

   ```
   CRTJRN JRN(QSYS/QPFRADJ) JRNRCV(QSYS/QPFRADJ1)
   ```

   You must use the name QSYS/QPFRADJ, and you must have authority to add objects to QSYS. Specify the name of the journal receiver you created in the previous step and any other options on the command.

Note: If dynamic tuning is active (system value QPFRADJ is 2 or 3), there is a slight delay in logging entries to the journal. If you need to start logging immediately, change the system value QPFRADJ to 1, and then change it back to 2 or 3. This will stop and restart the dynamic tuning support.

After you create the performance journal, use the following steps to copy the performance changes to a file:

1. Create a file of your choice by using the Create Duplicate Object (CRTDUPOBJ) command:

   ```
   CRTDUPOBJ OBJ(QAWCTPJE) FROMLIB(QSYS)
             OBJTYPE(*FILE) TOLIB(MYLIB)
             NEWOBJ(MYFILE)
   ```

   The newly created file is formatted to match the format of the journal entries.

2. Copy the journal entries to the file by using the Display Journal (DSPJRN) command:

   ```
   DSPJRN JRN(QSYS/QPFRADJ) ENTTYP(TP)
          OUTPUT(*OUTFILE) OUTFILE(MYLIB/MYFILE)
   ```

Figure 13-18 lists the various fields (found in file QSYS/QAWCTPJE) in the performance tuning (TP) journal entry:

Figure 13-18. Journal Entry Formats

Field Name	Description	Field Attributes
TPPNAM	Name of shared pool	Character(10)
TPFLG1	Pool changed flag	Character(1)
TPCSIZ	Current pool size	Packed decimal(8,0)
TPCRES	Pool reserve size	Packed decimal(8,0)
TPCACT	Current activity level	Packed decimal(6,0)
TPDFLT	Database page faults per second	Packed decimal(6,2)
TPNFLT	Nondatabase page faults per second	Packed decimal(6,2)
TPWI	Job transitions from wait to ineligible	Packed decimal(6,0)
TPAW	Job transitions from active to wait	Packed decimal(6,0)
TPCJOB	Current number of jobs running in pool	Packed decimal(6,0)
TPAJOB	Average number of jobs running in pool	Packed decimal(6,0)
TPNSIZ	New pool size	Packed decimal(8,0)
TPNACT	New activity level	Packed decimal(6,0)
TPVMIN	VLIC minimum	Packed decimal(8,0)
TPAI	Active to ineligible	Packed decimal(6,0)

Dynamic Storage Pool Paging

Dynamic tuning using storage pool paging can improve performance depending on how it brings objects into memory.

- If many jobs are running in a small storage pool, the system limits the amount of memory used by each job.
- If the storage pool for those jobs has enough memory, the system determines (object by object) how much data to bring into the storage pool.
- If objects are referred to sequentially, the system brings larger blocks of data into memory and delays writing changes of the data. This reduces the number of I/O operations issued by the job and reduces the contention for disk drives, which, in turn, reduces the time that jobs wait on I/O requests.
- If objects are referred to randomly, the system does not bring in large blocks of data because that does not reduce the number of I/O operations.

If you tune using dynamic paging and the system abnormally ends, the recovery time could be longer than the recovery time would be with fixed paging. See storage pool paging in the *Database Guide* or the QWCCHGTN API in the *System Programmer's Interface Reference* for ways to reduce this recovery time.

Chapter 14. Output

You can create output queues for particular uses such as by department and form types. The system automatically creates an output queue when you create a printer device (the same name is used). The parameters on the Create Output Queue (CRTOUTQ) command specify (parameter names are given in parentheses):

- If users that have authority to read the output queue can display or copy output data from files other than their own (DSPDTA)
- How many, if any, job separators (pages) are to be made between each job's output when written to the output device (does not apply to diskette) (JOBSEP)
- If a user having job control authority can control the output queue and its contents (OPRCTL)
- If files should be ordered on the queue by job start time or spooled file creation time (SEQ)
- Authority (AUT)
- Text description (TEXT)

For a list of the spooling commands, type `go cmdspl` from the command line. For more detailed information on printed output, see the *Guide to Programming for Printing*.

Job and Output Queues for Spooling

Spooling is supported in a manner similar to batch jobs. QSPL is started from the shipped subsystem description QSYS/QSPL. QSPL supports the processing of spooling readers and writers that transfer data to and from devices independently of the application processing. Figure 14-1 shows how the spooling subsystem QSPL operates.

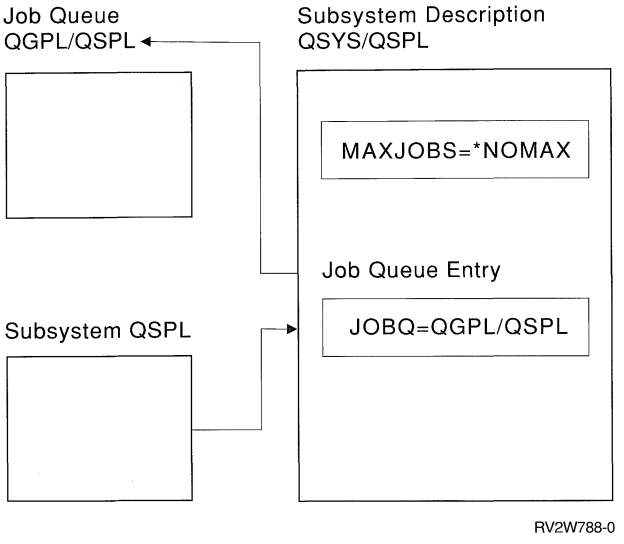


Figure 14-1. QSPL Spooling Subsystem Operation

When the subsystem QSPL is started, it processes the jobs on the queue QGPL/QSPL until it is empty or the subsystem is ended.

Because a separate job description is used for each type of reader or writer, you can set up your system to uniquely handle different types of spooling processing. See the section on IBM-supplied objects in Appendix B, "IBM-Supplied Object Contents," for a description of the spooling job descriptions.

How To's

This section tells you how to perform some common tasks involving creating a programmer output queue.

Creating a Programmer Output Queue

You may want to create a special output queue so your spooled output is always sent to that queue. For example, you could allow the programmer to use the Work with Spooled File (WRKSPLF) command or the source entry utility (SEU) to display the spooled output at a work station and then selectively print or delete the spooled output.

First, use the Create Output Queue (CRTOUTQ) command to create the output queue and the Grant Object Authority (GRTOBJAUT) command to give the programmer authority to the output queue:

```
CRTOUTQ OUTQ(QGPL/PGMR) AUT(*NONE)
        AUTCHK(*DTARIGHTS)
GRTOBJAUT OBJ(PGMR) OBJTYPE(*OUTQ) USER(QPGMR)
        AUT(*USE)
```

To use this new output queue, you could change the OUTQ parameter in the QPGMR user profile, or you could do one of the following:

- For interactive jobs, you can change the initial program for the programmer so the new initial program specifies this output queue:

```
PGM
CHGLIBL LIBL(TESTLIB QGPL QTEMP)
CHGJOB OUTQ(PGMR)
TFRCTL QSYS/QPGMMENU
ENDPGM
```

- Other functions, such as the logging level for messages, could also be set in this initial program. The libraries specified on the Change Library List (CHGLIBL) command must exist on the system. Generally, the first library specified contains the basic programmer objects such as source files and a job description.
- Another alternative is to assign a unique job description to the user profile that contains the parameter OUTQ(PGMR).
- For batch jobs, you create a job description with the Create Job Description (CRTJOBD) command and specify the QGPL/PGMR output queue on the OUTQ parameter.

Chapter 15. Job Accounting

This chapter describes the job accounting function which gathers data so that you can determine who is using your system and what system resources they are using. It also assists you in evaluating the overall use of your system.

Typical job accounting data details the jobs running in your system and the resources they are using such as the use of the processing unit, printer, display stations, database and communications functions.

Job accounting is optional. You must take specific steps (described later in the section "Setting Up Job Accounting" on page 15-10) to set up job accounting. You may also assign accounting codes to user profiles or specific jobs.

Job accounting statistics are kept by using the journal entries made in the system accounting journal QSYS/QACGJRN. You should know how to perform journal management operations, such as saving a journal receiver, changing journal receivers, and deleting old journal receivers. For more information about journal management, see the *Advanced Backup and Recovery Guide*.

When you want to analyze the job accounting data, it must be extracted from the QACGJRN journal by use of the DSPJRN command. With this command you can write the entries into a database file. You must write application programs or use a utility such as the query utility to analyze the data.

You may request the system to gather job resource accounting, printer file accounting data, or both. The system provides different journal entries for each type:

- Job resource accounting: The job (JB) journal entry contains data summarizing the resources used for a job or for different accounting codes used in a job.
- Printer file accounting:
  - Direct print (DP) journal entry: Contains data about printer files produced on print devices (nonspooled).
  - Spooled print (SP) journal entry: Contains data about printer files made by a print writer (spooled).

Resource Accounting

Resource accounting data is summarized in the JB journal entry at the completion of a job. In addition, the system creates a JB journal entry summarizing the resources used each time a Change Accounting Code (CHGACGCDE) command occurs. The JB journal entry includes:

- Fully qualified job name
- Accounting code for the accounting segment just ended
- Processing unit time
- Number of routing steps
- Date and time the job entered the system
- Date and time the job started
- Total transaction time (includes service time, ineligible time, and active time)
- Number of transactions for all interactive jobs
- Auxiliary I/O operations
- Job type
- Job completion code
- Number of printer lines, pages, and files created if spooled or printed directly
- Number of database file reads, writes, updates, and deletes
- Number of ICF file read and write operations

Some of the job accounting information can also be accessed using the CPF1124 and CPF1164 messages located in the QHST log. Job accounting data available through the QHST messages is a subset of the method described here. For more information on QHST messages, see the *CL Programmer's Guide*. See "Deciding Whether to Use Job Accounting" on page 15-4 to determine the method to use.

Printer File Accounting

There are two types of journal entries for printer file accounting:

- DP for nonspooled printer files
- SP for spooled printer files

These two types of journal entries share a common journal entry format although some of the information is only available in the SP entry. The DP and SP journal entries include information such as:

- Fully qualified job name.
- Accounting code.
- Device file name and library.
- Device name.
- Device type and model.
- Total number of pages and lines printed. If multiple copies occurred, this is the sum of all copies.
- Spooled file name (only in the SP entry).
- Spooled file number (only in the SP entry).
- Output priority (only in the SP entry).
- Form type (only in the SP entry).
- Total number of bytes of control information and print data sent to the printer device. If multiple copies occurred, this is the sum of all copies. (This only applies to the SP entry.)

The DP and SP journal entries occur when the file is printed. If a spooled file is never printed, no SP journal entry will appear.

Overview of Job Accounting

Figure 15-1 on page 15-3 shows an overview of job accounting as provided on the system.

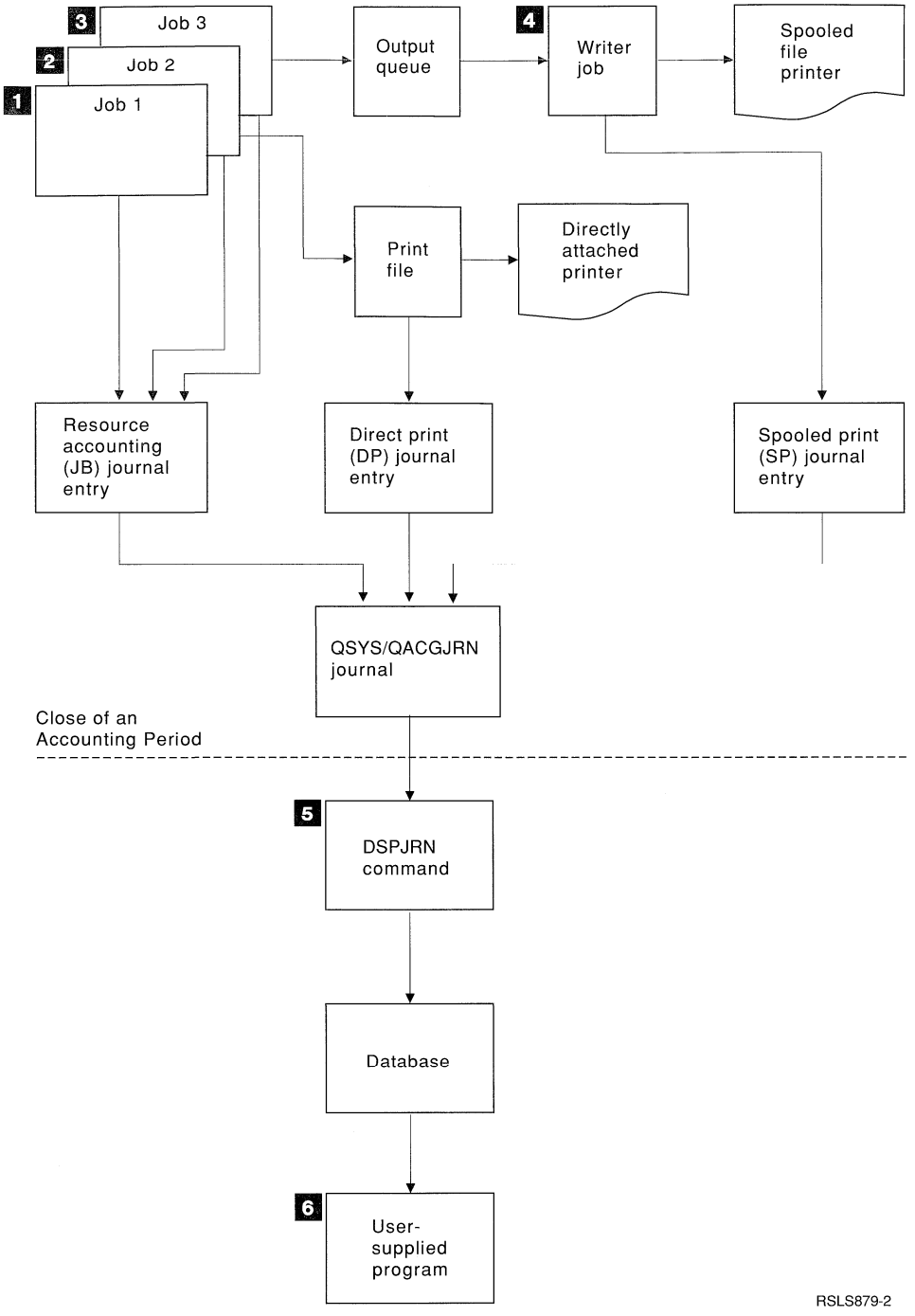


Figure 15-1. Job Accounting Overview

1 When job 1 is completed, the system summarizes the resources used and writes the JB journal entry to the QACGJRN journal. If the accounting code was changed during the job, a JB journal entry would be written for each time the accounting code was changed and at the end of the job. Job 1 does not make any printer output, and no job log is made. Therefore, no DP or SP journal entries are made for job 1.

2 Job 2 is printing a file directly to a printer. When the file is completed, a DP journal entry is written summarizing the printed data. When job 2 is completed, the system summarizes the resources used and writes the JB journal entry. Job 2 does not make any spooled printer output and no job log is made. Therefore, no SP journal entry is made for job 2.

3 Job 3 is printing to a file that is spooled. The SP journal entry is not written unless a print writer prints the file. When job 3 is completed, the system summarizes the resources used and writes the JB journal entry. If a job log is made at the completion of the job, it is a normal spooled file and an SP journal entry is created if the file is printed.

4 A print writer is started to print the files made by one or more jobs. When the writer finishes a file, it makes an SP journal entry. The SP journal entry is not made if the file is canceled before printing starts.

5 At the close of an accounting period (or whenever desired), the DSPJRN command can be used to write the accumulated journal entries into a database file.

6 User-written programs or the query utility can be used to analyze the accounting data. Reports such as resources used would compile data by a specific:

- Accounting code
- User
- Job type

Job Accounting Operating Characteristics

The AS/400 system attempts to allocate main storage as efficiently as possible. A job may not use the same amount of resources each time it is run. For example, if there are several active jobs on your system, a job spends more time reestablishing the resources needed for running than if a dedicated system environment is used. The system uses the job and run priorities assigned to different jobs to assist in managing main storage. Therefore, high priority jobs use less system resource than low priority jobs.

Because of these system operating characteristics, you may want to apply your own interpretation or algorithm to the job accounting data collected. If you are charging for the use of your system, you may want to charge more for:

- High priority jobs
- Work done during peak system time
- Use of critical resources

Deciding Whether to Use Job Accounting

Because the QHST messages (CPF1124 and CPF1164) are always available in the QHST log, the first question to answer is: Which method should I use to gather job accounting statistics? The following guidelines should help you answer the question.

Job accounting has all the information supplied by CPF1164 plus:

- Accounting code
- Number of print files, lines, and pages created by programs
- Number of database read, write, and update operations
- Number of communications read and write operations
- Actual lines and pages printed
- Time the job was active and suspended
- Actual number of bytes of control information and print data sent to the printer

The job accounting function is more effective for gathering job accounting statistics if:

- The resource information regarding database, printer, and communications use is important.
- Accounting codes are assigned to users or jobs.
- The information for printed output is important.
- Job accounting must be done on an accounting segment basis in a job rather than on a complete job basis.
- The active and suspended time information is needed.

The QHST messages are more effective for gathering job accounting statistics if:

- You do not want to manage the additional objects included in journaling.
- You do not need any resource information other than that provided in the CPF1124 and CPF1164 messages, which are sent automatically to the QHST log.
- You do not need print accounting information.

Some statistics recorded in the CPF1164 message and the JB journal entries will not match

exactly. This is due mainly to two factors: CPF1164 statistics are recorded slightly before the JB journal statistics; and each time an accounting code is changed, rounding occurs for some fields, while rounding occurs only once for CPF1164 messages.

Accounting Codes

The initial **accounting code** (up to 15 characters in length) for a job parameter is determined by the value of the ACGCDE (accounting code) parameter in the job description and user profile for the job. When a job is started, a job description is assigned to the job. The job description object contains a value for the ACGCDE parameter. If the default of *USRPRF is used, the accounting code in the job's user profile is used. Note, however, that when a job is started using the Submit Job (SBMJOB) command, its accounting code is the same as that of the submitter's job. You can change the accounting code after the job has entered the system by using the Change Accounting Code (CHGACGCDE) command.

The CRTUSRPRF and CHGUSRPRF commands support the ACGCDE parameter. The default is *BLANK. If all work for a particular user is to be recorded under one accounting code, only user profiles need to be changed. You can change the accounting codes for specific job descriptions by specifying the desired accounting code for the ACGCDE parameter on the CRTJOBD and CHGJOBD commands. The CHGACGCDE command also allows different accounting codes in a single job. You can develop a method of verifying the assignment of accounting codes as described in "Validity Checking of Accounting Codes" on page 15-15.

The Retrieve Job Attributes (RTVJOBA) command allows you to access the current accounting code in a CL program.

Resource Accounting Data

A JB journal entry is written at the end of every job and any time the job accounting code is changed by the CHGACGCDE command. A JB journal entry is written even if the accounting code is changed while the job is on the job queue although no resources have been used. Each resource accounting journal entry contains information about the resources used while the previous accounting code was in effect.

For example, Figure 15-2 illustrates a job with two accounting segments.

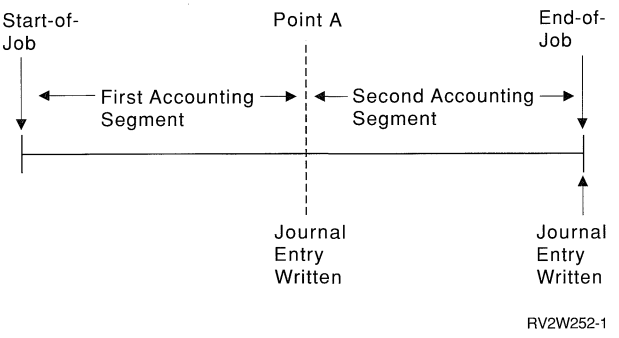


Figure 15-2. Job with Two Accounting Segments

At point A, the CHGACGCDE command was issued. The accounting code is changed and the JB journal entry is sent to the journal. The JB journal entry contains data for the first accounting segment. When the job ends, a second JB entry is made for the job containing data for the second accounting segment.

If the job accounting code was not changed during the existence of the job, the single JB entry summarizes the total resources used by the job. If the job accounting code was changed during the existence of the job, then you must add up the fields in the multiple JB entries in order to determine the total resources used by the job. The creation of a job log does not count toward the processing unit use for a job or its printed output in the JB accounting entries. However, if you are using print file accounting, the job log printed is included in the printer file journal entries.

General Accounting Journal Information

Each journal entry contains the standard prefix fields for any journal entry (for example, date, time, journal sequence number). See the *Advanced Backup and Recovery Guide* for a discussion of these fields.

JB Accounting Journal Information

Figure 15-3 on page 15-6 lists the various fields (found in field reference file QSYS/QAJBACG) in the JB journal entry.

Figure 15-3. Fields Found in JB Journal Entry

Field Name	Description	Field Attributes	Comments
JAJOB	Job name	Character (10)	
JAUSER	Job user	Character (10)	
JANBR	Job number	Zoned (6,0)	
JACDE	Accounting code	Character (15)	
JACPU	Processing unit time used (in milliseconds)	Packed decimal (11,0)	See Note 1 on page 15-7
JARTGS	Number of routing steps Job entered the system	Packed decimal (5,0)	
JAEDTE	–Job entry date (mmddyy format)	Character (6)	
JAETIM	–Job entry time (hhmmss format) Job start date and time	Character (6)	See Note 2 on page 15-7
JASDTE	–Job start date (mmddyy format)	Character (6)	
JASTIM	–Job start time (hhmmss format)	Character (6)	
JATRNT	Total transaction time (in seconds)	Packed decimal (11,0)	See Note 3 on page 15-7
JATRNS	Number of transactions	Packed decimal (11,0)	See Note 4 on page 15-7
JAAUX	Auxiliary I/O operations and database operations (including page faults for any reason)	Packed decimal (11,0)	
JATYPE	Job type	Character (1)	See Note 5 on page 15-7
JCCDE	Completion code	Packed decimal (3,0)	See Note 6 on page 15-7
JALINE	Number of print lines	Packed decimal (11,0)	See Note 7 on page 15-7
JAPAGE	Number of printed pages	Packed decimal (11,0)	
JAPRTF	Number of print files	Packed decimal (11,0)	
JADBPT	Number of database write operations	Packed decimal (11,0)	See Note 8 on page 15-7
JADBGT	Number of database read operations	Packed decimal (11,0)	See Note 8 on page 15-7
JADBUP	Number of database update, delete, FEOD, release, commit, and rollback operations	Packed decimal (11,0)	See Note 8 on page 15-7
JACMPT	Number of communications write operations	Packed decimal (11,0)	See Note 9 on page 15-7

Figure 15-3. Fields Found in JB Journal Entry

Field Name	Description	Field Attributes	Comments
JACMGT	Number of communications read operations	Packed decimal (11,0)	See Note 9 on page 15-7
JAACT	Time job was active (in milliseconds)	Packed decimal (11,0)	
JASPN	Time job was suspended (in milliseconds)	Packed decimal (11,0)	

Notes:

1. The processing unit time does not include processing unit use and printer statistics for the creation of job logs.
2. For job completion date and time from journal entries, use the JODATE and JOTIME fields that are part of the standard journal entry prefix information. (See the *Advanced Backup and Recovery Guide*, for more information on these fields.) After an abnormal system ending, these fields contain the current date and time and not (as with CPF1164 messages) the actual time of the system ending.
3. The total transaction time is set to -1 when:
   - Time is set backward.
   - An overflow occurred in a file on a computation.
   - The system went down while the job was active.
4. The last transaction (SIGNOFF) is not counted.
5. The job types recorded are the following:

   A Autostart job
   B Batch job (includes communications and MRT)
   I Interactive job
   M Subsystem monitor
   R Spooling reader
   W Spooling writer

   Note: These are the same as those used in message CPF1164, except that message CPF1164 includes some system job information not included in the journal entries.
6. The completion codes, which are similar to those used for message CPF1164, are:

   000 Normal completion
   010 Normal completion during controlled end or controlled subsystem end
   020 Job exceeded end severity
   030 Job ended abnormally
   040 Job ended before becoming active
   050 Job ended while active
   060 Subsystem ended abnormally while job was active
   070 System ended abnormally while job was active
   080 Job completed in the time limit
   090 Job forced to complete after the time limit has ended
   099 Accounting entry caused by CHGACGCDE command
7. The number of print lines does *not* reflect what is actually printed. Spooled files can be canceled or printed with multiple copies. The information in the JB journal entry reflects only what was written by the program. This excludes any lines written for the job log. See the discussion on DP and SP printer file accounting data later in this chapter.
8. The numbers recorded for database I/O operations do not include I/O operations to readers and writers, or I/O operations caused by the CL commands CPYSPLF, DSPSPLF, or WRKSPLF. If SEQONLY(*YES) is in effect, these numbers show each block of records read, not the number of individual records read.
9. The numbers recorded for communications I/O operations do not include remote work station activity. When the I/O is for a communications device, the numbers include only activity related to ICF files.

DP and SP Printer File Accounting Data

The accounting code used for the DP or SP journal entries is the accounting code of the job at the time the file is closed. Sometimes a DP or SP entry is created before the file is closed (such as when a writer which is creating a SCHEDULE(*IMMED) file is ended). When this

happens the current accounting code of the job is used.

A DP or an SP journal entry is created for each file printed. If the job log is spooled and then printed, an SP entry is created for it. Also, an SP entry is written for diskette spooled files redirected to a printer by the print writer.

DP Accounting Journal Information:

Figure 15-4 on page 15-7 lists the various fields (found in file QSYS/QAPTACG) in the DP journal entry.

Figure 15-4. Fields Found in the DP Journal Entry

Field Name	Description	Field Attributes
JAJOB	Job name	Character (10)
JAUSER	Job user	Character (10)
JANBR	Job number	Zoned (6,0)
JACDE	Accounting code	Character (15)
JADFN	Device file name	Character (10)
JADFNL	Library in which device file is stored	Character (10)
JADEVN	Device name	Character (10)
JADEVT	Device type	Character (4)
JADEVM	Device model	Character (4)
JATPAG	Total number of print pages produced	Packed decimal (11,0)
JATLIN	Total number of print lines produced	Packed decimal (11,0)
JASPFN	Always blank	Character (10)
JASPNB	Always blank	Character (4)
JAOPTY	Always blank	Character (1)
JAFMTP	Always blank	Character (10)
JABYTE	Always zero	Packed decimal (15,0)
JAUSRD	User data	Character (10)

SP Accounting Journal Information:

The information provided is similar to that provided in the DP accounting journal data except that the spooled file name, spooled file number, output priority, form type, and total number of bytes of control information and print data sent to the printer are included.

An SP journal entry is not written if a spooled file is deleted before a writer starts writing the file to the device.

Figure 15-5 on page 15-8 lists the fields (found in file QSYS/QAPTACG) in the SP journal entry.

Figure 15-5 (Page 1 of 2). Fields Found in SP Journal Entry

Field Name	Description	Field Attributes	Comments
JAJOB	Job name	Character (10)	
JAUSER	Job user	Character (10)	
JANBR	Job number	Zoned (6,0)	
JACDE	Accounting code	Character (15)	
JADFN	Device file name	Character (10)	
JADFNL	Library in which device file is stored	Character (10)	
JADEVN	Device name	Character (10)	
JADEVT	Device type	Character (4)	
JADEVM	Device model	Character (4)	
JATPAG	Total number of print pages produced	Packed decimal (11,0)	See Notes
JATLIN	Total number of print lines produced	Packed decimal (11,0)	See Notes
JASPFN	Spooled file name	Character (10)	
JASPNB	Spooled file number	Character (4)	
JAOPTY	Output priority	Character (1)	
JAFMTP	Form type	Character (10)	
JABYTE	Total number of bytes sent to the printer	Packed decimal (15,0)	See Notes
JAUSRD	User Data	Character (10)	

Notes:

1. The system attempts to record the actual number of pages, lines, and bytes printed, but when a writer is canceled *IMMED or recovers from a device error (such as end of forms), it is not possible to determine the exact number of pages, lines, and bytes printed.
2. Extra pages and lines produced with the alignment line are not included in the page, line, and byte counts.
3. If a spooled file goes into WTR status (but is set to MSGW) or if the file is deleted while in MSGW status, an SP journal entry will appear in the DP accounting journal indicating that there are 0 pages and 0 lines printed.
4. While using a printer configured AFP(*YES), if you delete or hold a file immediately after it has printed pages, the SP entry for that file may indicate 0 pages and 0 lines printed although some pages were printed.
5. The page, line, and byte counts for the job and file separators are included with the counts for the file they are associated with.
6. When an IPDS file contains graphics or bar codes and is sent to an IPDS printer that does not support graphics or bar codes, the page, line, and byte counts include the unprinted graphics and bar codes.
7. If printer configuration is AFP(*YES), the field for total number of print lines produced is zero. The total number of pages produced field is correct.

Batch Processing

Any batch job submitted during the running of a job using the SBMJOB command automatically uses the same accounting code as the job that submitted the batch job. When the SBMJOB command is used, the accounting codes cannot be overridden regardless of how the job description entry is coded. If you want the batch job to operate under an accounting code other than that of the submitting jobs, a CHGACGCDE command should be issued either:

- Before and after the SBMJOB command is issued
- Immediately by the batch job

Batch jobs submitted using a reader or a SBMDBJOB or SBMDKTJOB command use the accounting code specified in the job description for the batch job. If the job description specifies ACGCDE(*USRPRF), the accounting code is taken from the user profile used for the job.

Interactive Processing

If an interactive job has a fixed set of options for a user and each option has an assigned accounting code, it may be desirable to automatically assign a new code when the user requests to work on a new function. A typical approach would be for a menu option to request a new functional area.

The CHGACGCDE command would then be issued within a CL program and the job values used for the previous accounting code would be summarized in the JB accounting journal entry.

If a user has several assignments for which only he knows the accounting codes to be used, you can:

- Give authority to the user to enter the CHGACGCDE command.
- Write a program to prompt the user for the accounting code.

Note: For source pass-through jobs, the job accounting information does not include the target pass-through job. For target pass-through jobs, the job accounting information does not include the associated communications batch job.

System Job Processing

System jobs that you control (for example, readers and writers) are assigned an accounting code of *SYS. Other system jobs that you do not control (for example, QSYSARB, QLUS, SCPF) do not receive a journal entry.

Note: You cannot use the CHGACGCDE command to change the accounting code of the subsystem monitor or a reader or writer. You can, however, change the accounting code of a reader or writer by changing the appropriate IBM-supplied job descriptions and user profiles and then starting them again.

Setting Up Job Accounting

To set up resource or printer file accounting, perform the following:

1. Create a journal receiver in a library of your choice by using the Create Journal Receiver (CRTJRNRCV) command:

   ```
   CRTJRNRCV  JRNRCV(USERLIB/ACGJRN1)
   ```

   You should name the journal receiver ACGJRN1 or a similar name that can be used to create a naming convention such as ACGJRN2, ACGJRN3, for future journal receivers.

   After you create the first receiver, you can create additional receivers and attach them to the QSYS/QACGJRN journal automatically with the correct naming convention by using the CHGJRN JRNRCV(*GEN) command. You will probably want to place the journal receiver in one of your libraries that is saved regularly.

   You can specify options on the CRTJRNRCV command to assist in your operational aspects. If you want to use dual journal receivers, you must create a second journal receiver.

2. Create the journal QSYS/QACGJRN by using the Create Journal (CRTJRN) command. The name QSYS/QACGJRN *must* be used, and you must have authority to add objects to QSYS. You need to specify the name of the journal receiver(s) you created in the previous step and any other options on the command. You should also consider who you want to have authority to this journal.

3. As the security officer, change the accounting level system value QACGLVL using the Work with System Value (WRKSYSVAL) command or the Change System Value (CHGSYSVAL) command. When the system value is changed, any new jobs started on the system will automatically produce a job accounting journal entry when the jobs are completed.

   The VALUE parameter on the CHGSYSVAL command determines when job accounting journal entries are produced:

VALUE Parameter	**Description**
*NONE	The system does not produce any entries in the job accounting journal.
*JOB	The system produces a JB journal entry for each accounting segment of a job.
*PRINT	The system produces a DP or an SP journal entry for each file printed (either a non-spooled file or a spooled file written by a print writer).
*JOB *PRINT	The system produces both types of journal entries.

   The system requires that the QSYS/QACGJRN journal be created before this system value is changed to request job accounting.

Note: If you specify Yes to save job accounting information about completed printer output using the Operational Assistant user interface, the QACGLVL system value is changed to *PRINT or *JOB *PRINT if job accounting is already on. A journal receiver and a journal are also automatically created, eliminating the need to manually create them. If you change the system value later to turn job accounting off, the completed printer output option on the Operational Assistant menu will not be available.

For information on processing the accounting journal entries, see the section "Converting Job Accounting Journal Entries" on page 15-13.

Note: If you need to record files in a journal that users have received and printed, you must first delete all the QPRTJOB system jobs for those users. End the QPRTJOB system job using the End Job (ENDJOB) command and option SPLFILE(*YES). This deletes all spooled files sent to this user using the Send Network Spooled Files (SNDNETSPLF) command and ends the user's QPRTJOB system job. The system value QACGLVL should be changed to *PRINT or *PRINT *JOB. The next time a file is sent to a user, a QPRTJOB job is created for that user and the job accounting level is correct for any files sent to that user.

Accounting Journal

The accounting journal QSYS/QACGJRN is processed as any other journal. Files can also be recorded in this journal although for simplicity it is recommended that you keep it solely for accounting information. You can use the SNDJRNE command to send other entries to this journal. While there are additional operational considerations involved in using several journals, there are advantages to *not* allowing any file entries in the QACGJRN journal. It is usually easier to control the QACGJRN journal separately so that all job accounting entries for a particular accounting period are in a minimal number of journal receivers and that a new journal receiver is started at the beginning of an accounting period. System entries also appear in the journal QACGJRN. These are the entries with a journal code of J, which relate to IPL and general operations performed on journal receivers (for example, a save of the receiver).

Job accounting entries are placed in the journal receiver starting with the next job that enters the system after the CHGSYSVAL command takes effect. The accounting level of a job is determined when it enters the system. If the QACGLVL system value is changed after the job is started, it has no effect on the type of accounting being performed for that job. The DP and SP print entries occur if the job that created the file is operating under accounting and the system value is set for *PRINT. If spooled files are printed after the accounting level has been set to *PRINT or if the job that created the file was started before the accounting level was changed, no journaling is done for those spooled files.

If damage occurs to the journal or to its current receiver so that the accounting entries cannot be journaled, a CPF1302 message is sent to the QSYSOPR message queue, and the accounting data is written to the QHST log in the CPF1303 message. The job trying to send the journal entry continues normally. Recovery from a damaged journal or journal receiver is the same as for other journals. See the *Advanced Backup and Recovery Guide* for recovery considerations.

The journal QACGJRN should not be allocated by another job. If the journal is allocated by another job, the journal entry is changed to message text and sent to the QHST log as message CPF1303.

You can use the OUTFILE parameter on the Display Journal (DSPJRN) command to write the accounting journal entries to a database file that you can process.

You can also use the RCVJRNE command on the QACGJRN journal to receive the entries as they are written to the QACGJRN journal.

Analyzing Job Accounting Data

While the most obvious use for the accounting data is to charge users for system resources, you may want to analyze the data in other ways. Some methods of analyzing the job accounting information would be sequencing by the following fields or combinations of fields:

- Accounting code

- User
- Job type (for example, batch or interactive)
- Time of day

Each of these sequences could be processed by a separate logical file.

A typical statistical analysis of a job would be a summary which reports on the:

- Type of job run with a count of occurrences and totals on processing unit time used
- Interactions
- Average response time
- Auxiliary I/O and file counts by type of file

Other items you may want to analyze are:

- Job termination code
- User profile name
- All jobs that exceed a certain amount of processing unit time

Security Considerations

Only the security officer (or a program adopting his authority) or a user with *ALLOBJ or *SECADM authority can change the system value QACGLVL. The change takes effect when a new job enters the system. This restriction ensures that if job accounting is in effect and the security officer performs system IPL, an accounting entry is written for the security officer's job.

You can assign job accounting codes only if you have the authority to use the CRTUSRPRF, CHGUSRPRF or CHGACGCDE command. This restricts the use of accounting codes and provides a basis for validity checking any changes.

Only a user with the *SECADM special authority is allowed to use the CRTUSRPRF and CHGUSRPRF commands. However, the security officer can delegate this authority by creating a CL program, which allows another user to adopt the security officer's profile and change the ACGCDE parameter in the user profile. The CL program could then have authority to one or more individuals.

The ACGCDE parameter also exists in job description objects, but the user must have the authority to use the CHGACGCDE command to enter a value other than the default of *USRPRF. This command is supplied with *CHANGE authority.

If you allow a user to use the CHGACGCDE command, he can:

- Create or change the ACGCDE parameter in job descriptions. (Authority to create or change job descriptions is also required.)
- Change the accounting code in his current job.
- Change the accounting code of a job other than his own if he also has the *JOBCTL special authority.

You can provide additional security by using the CHGACGCDE command in a CL program, which adopts the program owner's authority. This allows the user who is running an external function to perform a security-sensitive function without having direct authorization to the CHGACGCDE command.

The accounting journal and its receivers are treated as any other journal objects from a security viewpoint. Refer to the *Advanced Backup and Recovery Guide*, for a discussion of the journal security considerations. You must decide what authorization should exist for the accounting journal and journal receiver.

Recovery Considerations

If a job ends abnormally, the final accounting entry is written and all previously written accounting entries appear in the journal.

If an abnormal system ending occurs, the following accounting data is lost to the last routing step or last end-of-accounting segment, whichever occurred most recently.

- Information on the number of lines and pages printed
- Number of files created
- Database put, get, and update operations
- Communications read and write operations
- Auxiliary I/O operations
- Transaction time
- Number of transaction fields
- Time active
- Time suspended

After an abnormal system ending, the job completion time in the journal will not be the same as that in the CPF1164 message. The message uses the time nearest to that of the system ending, but the job accounting journal entries are sent to the journal during IPL, and the job completion time is the current system time, which is later than the time when the abnormal system ending occurred.

If the system ends abnormally, some journal entries can be lost. These are the entries that are written to the journal but not forced to disk (this is equal to FORCE(*NO) on the SNDJRNE command). They include the following:

- JB entries caused by a CHGACGCDE command
- DP and SP entries

Whenever a job completes, the last accounting code entry is forced to disk (as if FORCE(*YES) were specified on the SNDJRNE command). Whenever an accounting entry is forced to disk, all earlier entries in the journal, regardless of which job produced them, are forced to disk.

Exception: If only *PRINT accounting is specified on the system, there will not be any job ending FORCE(*YES) journal entries done. Therefore, if a critical accounting entry is written by a CHGACGCDE command and you want to ensure it will not be lost in case of an abnormal system ending, you could issue a SNDJRNE command and specify the FORCE(*YES) option. If files are also to be journaled to the accounting journal, any database changes are always forced to the journal, and this causes all earlier accounting entries to also be forced.

If an abnormal system ending occurs or you change an accounting code of a job other than your own, the qualified job name in the first 30 bytes of the JARES field in the journal entry describe the system job that wrote the JB entry at the next IPL and not the job that used the resources. The JAJOB, JAUSER, and JANBR fields should be used for analysis purposes.

If the job accounting journal or journal receivers become damaged, the system continues to operate and to record accounting data in the history log. To recover from the journal or journal receiver damage, use the Work with Journal (WRKJRN) command. Refer to the *Advanced Backup and Recovery Guide* for more information about the use of this command. After recovering the damaged journal or journal receiver, change the system value QACGLVL to a value appropriate for your installation. (Unless you change the QACGLVL system value, the system does not record accounting information in the new journal receiver.)

If you need to access the information from the CPF1303 message you will need to create a high-level language program. In order to define records that match the CPF1303 message you will need to include the following fields:

System Time	Char (8)
Message Record Number	Bin (4)
Qualified Job Name	Char (26)
Entry Type (JB, DP, or SP)	Char (2)
Length of Data	Bin (2)

Followed by fields:

JAJOB through JASPN for JB entries
JAJOB through JABYTE for SP and DP entries

(These fields are described earlier in this chapter.)

For an example program, refer to the section in *CL Programmer's Guide*, that discusses processing the QHST file for the job completion message.

The CPF1164 message always consists of three records and the CPF1303 message always consists of four records.

The information contained in the standard journal prefix fields is not included in this message. All that is needed is information pertaining to the job end, date, and time; this information can be found in record 1 of the CPF1303 message.

Converting Job Accounting Journal Entries

You can use the OUTFILE parameter on the DSPJRN command to write the job accounting journal entries into a database file that you can process. The OUTFILE parameter allows you to name a file or member. If the member exists, it is cleared before the records are written. If the member does not exist, it is added. If the file does not exist, a file is created using the record

format QJORDJE. This format defines the standard heading fields for each journal entry, but the job accounting data is defined as a single large field.

To avoid having to process the accounting data as a single large field, two field reference files are supplied to help you in processing the job accounting journal entries. The file QSYS/QAJBACG contains the record format QWTJAJBE and is used for JB entries. File QSYS/QAPTACG contains record format QSPJAPTE and is used for DP or SP entries. The same format is used for all printer file entries regardless of if the output is SP (spooled) or DP (nonspooled). The DP entry for directly printed files contains some fields that are not used; these fields contain blanks.

The following are some approaches you might use:

- The basic JB entries and DP or SP entries can be processed by creating two output files using the supplied field reference file formats and running the DSPJRN command once for JB and once for DP or SP. This allows you to define a logical file over the two physical files and use an high-level language program to process the externally described file. This solution is recommended and described later in this section.
- You can process only the JB entries by creating a file using one of the supplied field reference files (QSYS/QAJBACG) to create an externally described file. This file can then be processed by the query utility or an high-level language program.
- You can convert both types of journal entries using the default DSPJRN format of QJORDJE. You would then use a program-described file to process the journal entries in a high-level language program.

The following DDS defines a physical file for the JB journal entries using the QAJBACG field reference file in QSYS. You would normally create the file (using the CRTPF command) with the same name (QAJBACG) as the model file.

```
R QWTJAJBE    FORMAT(QSYS/QAJBACG)
```

The following DDS defines a physical file for the DP or SP journal entries using the QAPTACG field reference file in QSYS. You would normally create the file (using the CRTPF command) with the same name (QAPTACG) as the model file.

```
R QSPJAPTE    FORMAT(QSYS/QAPTACG)
```

You can specify a key field in either physical file; however, in this example, a logical file is used for sequencing.

If you create two physical files (one for JB and one for DP or SP) with the members of the same name, you would issue the following DSPJRN commands to convert the entries. Assume that you have created the physical files with the same names as the model files in your library YYYY.

```
DSPJRN  JRN(QACGJRN) JRNCDE(A) ENTTYP(JB)
        OUTPUT(*OUTFILE) OUTFILE(YYYY/QAJBACG)
DSPJRN  JRN(QACGJRN) JRNCDE(A) ENTTYP(SP DP)
        OUTPUT(*OUTFILE) OUTFILE(YYYY/QAPTACG)
```

You can control the use and selection criteria of the DSPJRN command so that you do not convert the same entries several times. For example, you can select all entries in a specific range of dates. You could convert all of the entries at a cutoff point for your job accounting analysis, for example, monthly. One or more journal receivers may have been used during the month. Note that each use of the DSPJRN command to the same member causes the member to be cleared before any new entries are added. Do not use the JOB parameter of the DSPJRN command as some entries are made for a job by a system job and will therefore not appear as you may expect them to.

Enter the following DDS to create a logical file to allow processing of both physical files. This allows you to read a single file in accounting code order and print a report using a high-level language program:

```
R QWTJAJBE    PFILE(YYYY/QAJBACG)
K JACDE
R QSPJAPTE    PFILE(YYYY/QAPTACG)
K JACDE
```

If you want to use a logical file to process only the basic job accounting record in accounting code order by a user name, you would enter the following DDS for a logical file:

```
R QWTJAJBE    PFILE(YYYY/QAJBACG)
K JACDE
K JAUSER
```

This logical file can be processed by the query utility or by a high-level language program.

If an abnormal system ending occurs, the qualified job name in the first 30 bytes of the JARES field in the journal entry describe the system job that wrote the entry at the next IPL and not the job that used the resources. For this reason, any analysis done on the JB entries should use the JAJOB, JAUSER, and JANBR fields.

Job Accounting Approaches

The following approaches may be useful to you in the setting up and managing your job accounting function.

Validity Checking of Accounting Codes

An important aspect of any data processing application is ensuring that the correct control fields are specified. For job accounting codes, this can require a complex validity-checking function which not only checks for the existence of authentic codes, but also checks which users are allowed to use specific codes.

Accounting codes can be assigned in the following areas:

- User profile
- Job description
- In a job (CHGACGCDE command)

Controlling the Assignment of Accounting Codes in User Profiles

If it is important to control the assignment of accounting codes, you may want to consider the following approach. Before an accounting code is placed in a user profile, you may want to ensure that the code is valid and that it is valid for a particular user.

You could control the changing of accounting codes on the CHGJOBD command by giving only the security officer authority to the CHGACGCDE command, and therefore, another user could not create or change a job description with a specific accounting code.

The CHGACGCDE command could be used directly to allow users to change the job accounting code of their own or another job. To change another job, the user must also have the special authorization of *JOBCTL.

In many installations, it would not be valid to allow the change of an accounting code for a job on the job queue or for one job to change the accounting code of another job. Controlling this and ensuring validity checking of the new accounting code could be done with a CL program and command.

For example, the CHGACGCDE command would be privately authorized and included in a CL program where it only changed the current job (for example, JOB(*) is specified). The command would be authorized appropriately.

Appendix A. Performance Data

This chapter explains how to collect performance data, and how the collection occurs. It also contains reference information that shows the format and content of the database files collected. This information is provided so the user can collect performance data and write an application to analyze the data.

Reasons for Collecting Performance Data

You may choose to collect performance data for any of the following reasons:

- To do analysis with the IBM Performance Tools/400 licensed program
- To gather input for an analysis application that reduces the data into meaningful reports
- To do analysis on another system that has the IBM Performance Tools/400 licensed program installed

Preparing to Collect Performance Data

Before starting data collection, consider what type of data to collect, how often to collect it, and when to collect it.

Data collection may slightly degrade performance of the system due to the time involved in collecting the performance data. The amount of degradation is related to the amount of data collected, so you should only collect the data you need.

Three types of data can be collected:

- System data. This data consists of performance data relating to the following: all jobs on the system, devices attached to the system, storage pools, communications I/O processors, disk I/O processors, local work station I/O processors, and work station response times.
- Communications data. This data includes all of the previously listed system data and statistics for the following protocols: binary synchronous communications, asynchronous communications, X.25, Ethernet, token-ring network, and synchronous data link control (SDLC).
- Trace data. This data includes internal system trace data from the vertical microcode trace table. This type of data should not be collected unless it is going to be analyzed by the IBM Performance Tools/400 licensed program.

How to Collect Performance Data

This section explains how to start and end the collection of performance data.

Starting Performance Data Collection

The Start Performance Monitor (STRPFRMON) command starts performance data collection. Issuing this command submits a batch job to a job queue. When the performance monitor batch job is started from the job queue, data collection begins. The job continues to collect data until one of the following occurs:

- The time limit specified for data collection is reached.
- The End Performance Monitor (ENDPFRMON) command is issued.
- The performance monitor job is ended abnormally (End Job (ENDJOB), End Subsystem (ENDSBS), or End System (ENDSYS) command).

While the performance monitor job is active, the monitor periodically collects data and puts it in the performance database files. For more information on how data collection occurs, see "How Performance Data Collection Occurs" on page A-3. For more information on the Start Performance Monitor (STRPFRMON) command, see the *CL Reference* manual.

Collecting System Data Only: To collect only system data, specify *SYS for the DATA parameter on the STRPFRMON command.

Collecting System and Communications Data: To collect communications data, specify *ALL for the DATA parameter on the STRPFRMON command.

Note: Communications data is collected in addition to system data, not instead of it.

Collecting Trace Data: To collect trace data, you must specify a value other than *NONE for the TRACE parameter on the STRPFRMON command. This causes the performance monitor to start the trace (using the Trace Internal (TRCINT) command). Turning on the trace causes the system to log various activities into an internal trace table. Eventually, the trace data is dumped from the internal trace table into a database file (QAPMDMPT).

Dumping the trace data from the internal trace table into the trace database file also degrades performance. It is probably not a good idea to dump the trace during peak activity on a heavily loaded system. So when the TRACE parameter indicates to start the trace, you should also consider when the trace should be dumped to the database file.

The dump trace (DMPTRC) parameter on the STRPFRMON command allows you to specify if the dump should take place when the performance monitor ends. The DMPTRC parameter is also on the ENDPFRMON command, so that you can override what you specified for the DMPTRC parameter on the STRPFRMON command.

If the trace table is not dumped when the performance monitor ends, it can be dumped later with the Dump Trace (DMPTRC) command. This command allows dumping of the trace table (which may degrade system performance on heavily loaded systems) to be delayed until a time that would not affect other users. If the trace is not dumped when the performance monitor ends, the trace data remains in the internal system trace table. If the trace table is cleared for any reason (running another system trace clears the trace table), the performance trace data is lost. Therefore, it is a good idea to dump the trace table as soon as possible after the performance monitor ends.

Dumping Trace Data: The DMPTRC command is used to put information from an internal trace table into a database file. This command allows the dump to be done by the job that issued the command, or, by submitting a batch job to a job queue, to have the trace dumped by the batch job.

For more information on the DMPTRC command, see the *CL Reference* manual.

The Internal Trace Table: The internal trace table has a maximum size of 16MB of storage. Before starting any traces, the performance monitor sets the trace table size *limit* to this maximum value to allow the most information possible to be collected. The trace table can use up to 16MB, but is not initially set to that value. If the trace is not started, the size of the trace table is not changed.

After the performance monitor starts the trace, the system continues to log trace data until either the trace table fills up or the trace stops (which is done automatically when the performance monitor ends). If the trace table fills up while the performance monitor is running (and the performance monitor started the system trace), a message is sent to the system operator's message queue and a user's message queue (if one was specified on the MSGQ parameter of the STRPFRMON command). The performance monitor then automatically turns off the trace. At that point, three options are available:

- Immediately dump the trace table to a database file (by using the DMPTRC command). Using this option ensures that the trace data is not written over if other traces are turned on. However, this approach may degrade performance on heavily loaded systems.
- Wait until the performance monitor ends and have the performance monitor dump the trace at that time (this is the default). If another trace is started before the performance monitor ends, the data in the internal trace table is written over, losing the trace information that the performance monitor produced.
- Do not dump the trace when the performance monitor ends. Instead, dump the trace at a convenient time with the DMPTRC command.

Ending Performance Data Collection

The ENDPFRMON command is used to end performance data collection. Issuing the command causes the performance monitor batch job to make a final collection of performance data and then end itself. This command should be used instead of the ENDJOB, ENDSBS, and ENDSYS commands if you want the final data collected. Only the ENDPFRMON command causes the final data to be collected.

For more information on the ENDPFRMON command, see the *CL Reference* manual.

Collecting Performance Data Automatically

You can choose to have your system automatically collect performance data on a weekly schedule. You can specify particular days of the week you want data to be collected.

To establish the automatic weekly collection of performance data, use the Work with Performance Collection (WRKPFRCOL) command. After entering the command, the Work with Performance Collection display is shown. You need to provide a name for the performance collection, the day of the week and time you want the collection to occur, and a description of the collection for your reference. You may also make optional changes to the parameters that are used for the Start Performance Monitor (STRPFRMON) command. The Work with Performance Collection display also allows you to add, change, remove, hold, release, or display a collection of performance data.

Use the Add Performance Collection (ADDPFRCOL) and Change Performance Collection (CHGPFRCOL) commands to add or change a performance collection.

Automatic performance collection requires a batch job (QPFRCOL) that queries the schedule created by the ADDPFRCOL command and submits the STRPFRMON command at the appropriate times. If you are using automatic performance collection for the first time, you need to submit the performance collection job as follows:

```
SBMJOB JOBD(QGPL/QPFRCOL) USER(*JOBD)
       RQSDTA(*JOBD) RTGDTA(*JOBD)
```

You must submit this job after using the ADDPFRCOL command. Type this exactly as indicated here; do not change the user, request data, or routing data in the job description. This batch job runs until all the performance collections are removed or held.

This job exists as an autostart job entry in the IBM-supplied subsystems QBASE and QCTL. If you are using the latest release of one of these controlling subsystems, then as long as you have performance collections defined, the batch job is started for you after each IPL.

If you are not using the latest release of QBASE or QCTL, you can add an autostart job entry for the performance collection to one of the subsystems. See Chapter 8, "Autostart Jobs," for more information about autostart job entries.

How Performance Data Collection Occurs

Performance data collection occurs when a batch job is submitted to a job queue with the STRPFRMON command, or the ADDPFRCOL command for automatic data collection. No data is collected until the job is started from the job queue, so an active job queue should be specified for the JOBQ parameter of the STRPFRMON command.

Although the performance monitor is a batch job, it runs at a higher priority than interactive jobs, but lower than system-level jobs.

After the performance monitor job is started, it opens the data collection database files and performs other initialization functions. After initialization, the performance monitor job waits for a data collection interval to occur.

When the performance monitor collects data, it retrieves performance counters from the hardware devices attached to the system. The counters that are stored in the database file for an interval indicate the amount of resources used for just that interval (not a summary of the total use since the performance monitor started).

The performance monitor job collects data for the amount of time specified on the STRPFRMON command. When the time limit is reached, the performance monitor puts the final performance counters into the database file (regardless of whether a complete interval occurred) and then ends itself. This means that if you specified a 60-minute interval and a 2-1/2 hour time limit, the database files contain entries placed in the files at 1 hour, 2 hours, and 2-1/2 hours.

The ending time for the performance monitor is accurate within 5 minutes of the time you specify. For example, if you started the performance monitor at 1:00 and specified it should end itself in two hours, the performance monitor will end between 2:55 and 3:00.

If the performance monitor was started and you want to end it before the time limit is reached, you can use the ENDPFRMON command. The ENDPFRMON command puts the final performance counters into the database files, and ends the performance monitor batch job (and, if specified, dumps the performance trace into a database file).

Internal Data Collection Intervals

A data collection interval is the amount of time during which the performance monitor job is active to retrieve data from the system.

Although you can specify that data is to be placed in the database files at intervals ranging from 5 to 60 minutes (the default is 15 minutes), counter limitations require the performance monitor to retrieve performance data from the system every 5 minutes. Some devices have counters that can wrap (when the counter reaches its maximum value, it is reset to zero). The performance monitor can handle one wrap when calculating the counts for the interval, but if the counter wraps twice before the performance monitor retrieves the counter, the data is lost for one wrap (there is no indication that the counter wrapped twice). The maximum amount of time the performance monitor can run before risking a double wrap is 5 minutes. Therefore, the performance monitor must retrieve the data every 5 minutes to ensure that there is no loss of data.

When the Data Is Collected

Regardless of the value specified for the collection interval, the performance monitor must collect counters from the devices attached to the system every five minutes. Any of these five-minute intervals that do not coincide with the user specified collection interval are called an *internal interval*. Any of the five-minute intervals that coincide with the user-specified collection intervals are called *database intervals*, because the performance monitor writes the counters collected for each user-specified interval to the performance database files.

The example in Figure A-1 on page A-5 shows the difference between an internal interval and a database interval (assume that 15 minutes was specified for the INTERVAL parameter for the STRPFRMON command).

During internal intervals, the only data that is collected is from the devices attached to the system. This data is saved in internal tables until a database interval occurs, when it is written to the database files.

During database intervals, all of the data specified for collection is retrieved from the devices and the system, the values for the database interval are calculated (remember each interval shows the amount of use for just that interval), and the counters are written to the performance database files.

It is also possible for the performance monitor to become active between intervals. Certain events on the system cause the performance monitor to be notified when they occur. When one of these events occurs, the performance monitor momentarily becomes active (usually just long enough to store information about the event in an internal table), and then waits for the next collection interval to occur.

One of these events occurs when a job ends on the system. The only time that the performance monitor puts data into the database file other than at a database interval is when a job ends on the system (an entry is then inserted for that job for the final amount of resources used since the last database interval).

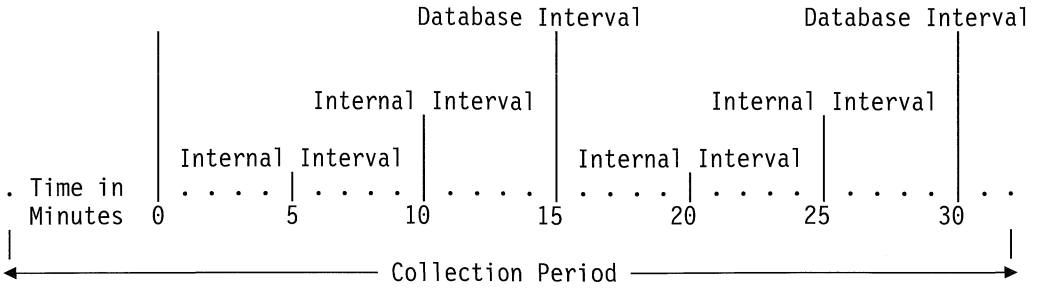


Figure A-1. Database Interval

Notification of Performance Monitor Status

While the performance monitor is active, it sends information messages to the system operator's message queue. These messages notify the system operator when the status of the performance monitor changes or an error occurs. Typical messages sent indicate that:

- The performance monitor was started (from the job queue).
- The performance monitor encountered an unexpected error.
- Another user issued the End Performance Monitor (ENDPFRMON) command.
- The performance monitor has ended.

Specifying the MSGQ parameter on the STRPFRMON command causes the performance monitor to send messages to the user's message queue whenever it sends messages to the system operator's message queue.

Database File Management

The performance monitor uses many different database files while collecting performance data, but the performance monitor places the data into the same member for all the files it uses. If the member specified on the STRPFRMON command does not exist when the performance monitor starts, it adds a member by that name to all the files it uses.

Because the default for the member name (MBR) parameter for the STRPFRMON command is *GEN (create the member name), it is possible to quickly add many members to the performance monitor database files. When using the default, you should periodically delete members in the performance database files when they are no longer used.

An alternative to deleting members is to delete the files themselves. If the performance monitor does not find a file in the specified library, it creates the file in that library and adds the specified member to the file.

Another alternative is to specify several member names over and over (if running every day of the week, name the members after the days of the week).

Estimating File Size for Collecting Performance Data: This section shows how to estimate the file size needed to collect system data and to collect system and communications data.

Collecting System Data Only: To estimate the amount of disk space for collecting only system data using the STRPFRMON command and specifying DATA(*SYS), add the following:

1500 plus the product of the number of intervals times the sum of:

2374	System data per interval
523	Times the average number of jobs active in the interval
243	Times the number of disk actuators
72	Times the number of pools
67	Times the average number of local work stations active in the interval
87	Times the average number of remote work stations active in the interval
178	Times the average number of communications controllers active in the interval
232	Time the number of multifunction controllers active in the interval

528 Times the average number of storage device controllers active in the interval

144 Times the average number of twinaxial work station controllers active in the interval

For example,

```
Disk space = 1500 + {#intervals x [(2374) +
(1865) (523 x avg #jobs) +
(1089 x APPC and Host CDs) +
(243 x #disk arms) + (72 x #pools) +
(67 x #ws) + (87 x #rws) +
(178 x #com ctl) + (1079 x avg #SNADS jobs) +
(232 x #mf IOP) + (528 x #dasd ctl) +
(144 x #wsc)]}
```

Collecting System and Communications Data: To estimate the amount of disk space for collecting both system and communications data using the STRPFRMON command by specifying DATA(*ALL), add the following:

1500 plus the product of the number of intervals times the sum of:

2374 System data per interval

523 Times the average number of jobs active in the interval

243 Times the number of disk actuators

72 Times the number of storage pools

67 Times the average number of local work stations active in the interval

87 Times the average number of remote work stations active in the interval

143 Times the average number of SDLC lines active in the interval

81 Times the average number of asynchronous lines active in the interval

207 Times the average number of binary synchronous communications (BSC) lines active in the interval

312 Times the average number of X.25 lines active in the interval

237 Times the average number of token-ring network (TRLAN) lines active in the interval

210 Times the average number of ELAN lines active in the interval

159 Times the average number of IDLC lines active in the interval

226 Times the average number of ISDN network interfaces active in the interval

178 Times the average number of communications controllers active in the interval

232 Times the average number of multifunction controllers active in the interval

528 Times the average number of storage device controllers

144 Times the average number of twinaxial work station controllers active in the interval

155 Times the average number of stations on Ethernet lines in the interval

155 Times the average number of stations on token-ring lines in the interval

73 Times the average number of SAPs per line times the average number of LAN lines in the interval

1865 APPN data per interval

1089 Times the number of APPC and host controller descriptions

| 107 Times the average number of SNADS jobs active in the interval

Content of Performance Data Files

The tables on the following pages show the format and content of performance data files collected by the system when using data collection commands. They also describe some of the relationships between the counters. The file names and descriptions used in the tables as follows:

Configuration Data (collected once per session):

File Name	Page	Description
QAPMCONF	A-8	System configuration data
QAPMSBSD	No field and byte data	Subsystem data

Performance Data (collected each interval):

File Name	Page	Description
QAPMSYS	A-9	System performance data
QAPMJOBS	A-26	Job data (one per job)
QAPMDISK	A-30	Disk storage data (one per read/write head)
QAPMPOOL	A-34	Main storage data (one per system storage pool)
QAPMHDLC	A-35	HDLC statistics (one per link)
QAPMASYN	A-37	Asynchronous statistics (one per link)
QAPMBSC	A-38	Binary synchronous statistics (one per link)
QAPMX25	A-40	X.25 statistics (one per link)
QAPMECL	A-42	Token-ring local area network statistics (one per link)
QAPMSTNL	A-44	Token-ring station file entries
QAPMETH	A-46	Ethernet statistics (one per link)
QAPMSTNE	A-48	Ethernet file station entries
QAPMDDI	A-49	Distributed Digital Interface (DDI) data
QAPMSTND	A-51	DDI station counter data
QAPMFRLY	A-53	Frame relay data
QAPMSTNY	A-54	Frame relay station
QAPMBUS	A-60	Bus counters (one per bus)
QAPMCIOP	A-61	Communications controller data (one per controller)
QAPMDIOP	A-64	Storage device controller data (one per controller)
QAPMLIOP	A-67	Twinaxial work station controller data (one per controller)
QAPMMIOP	A-62	Multifunction controller data (one per 9404)
QAPMRESP	A-69	Local work station response time (one per work station)
QAPMAPPN	A-82	APPN data
QAPMSNA	A-71	SNA data
QAPMRWS	A-70	Remote work station response time
QAPMSNADS	A-81	SNADS data
QAPMSAP	A-55	TRLAN, Ethernet, DDI, and Frame Relay SAP file entries
QAPMLAPD	A-56	Integrated services digital network LAPD file entries
QAPMIDLC	A-59	Integrated services digital network data link control file entries
QAPMLTG	No field and byte data	Licensed internal statistics

Trace Data:

File Name	Description
QAPMDMPT	System trace data (general description—no field and byte detail)
QAPMIOBS	Internal system observations (general description—no field and byte detail)
QAPMTSK	Internal transaction data (general description—no field and byte detail)

All the data except for QAPMCONF and QAPMSBSD is collected for each sample. QAPMCONF contains system configuration information that is reported only at the beginning of the performance monitor data collection job. QAPMSBSD contains subsystem data that is reported only at the end of the performance monitor data collection job.

Terms Used: The following abbreviations are used in the following tables.

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 communications 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.

File Name: QAPMCONF

System Configuration File Entries: Figure A-2 lists the various fields in the configuration file entries.

Figure A-2. System Configuration Data

Field Name	Description	Attributes	Buffer Position
GIOP	IOP BUS number/address. This field cannot be displayed (see Note 1).	C (1)	1
GUNIT	Reserved.	C (2)	2
GLVL	Resource level (see Note 3).	C (1)	4
GKEY	See Note 4 on page A-9.	C (2)	5
GDES	See Note 4 on page A-9.	C (10)	7
GDESCR	User-assigned physical line, local work station controller, and local work station names.	C (10)	17
GIOPB	IOP bus number.	PD (3,0)	27
GIOPA	IOP bus address.	PD (3,0)	29
GTYPE	Device or controller type.	C (4)	31
GMODEL	Device or controller model.	C (4)	35
GUNITA	The first digit of the 4-byte unit address.	PD (3,0)	39
GUNITB	The second digit of the 4-byte unit address.	PD (3,0)	41
GUNITC	The third digit of the 4-byte unit address.	PD (3,0)	43
GUNITD	The fourth digit of the 4-byte unit address.	PD (3,0)	45

Notes:

1. GIOP is a value only when the containing record is a configuration record. The first 3 bits are the bus number (value 0 through 2) and the remaining 5 bits are the bus address (value 0 through 31). Field cannot be displayed.
2. GUNIT is a value only when the containing record is a configuration record. Field cannot be displayed.
3. GLVL =

   1–Resource is an IOP
   2–Resource is an I/O adapter or controller
   3–Resource is a port or device

   GLVL is blank when the previous two fields (GIOP and GUNIT) are blank.

4.

GKEY	GDES
1	Performance monitor start date (yy/mm/dd/c).
2	Performance monitor start time (hh:mm:ss).
3	Model number (character 4 followed by 6 blanks).
4	Main storage size in KB (character 7).
5	Communications data collected (Y/N).
6	Machine serial number (character 8).
7	First response time boundary (zoned (10,0)) in milliseconds. The first response time monitor bracket is from 0 up to and including the first response time boundary.
8	Second response time boundary (zoned (10,0)) in milliseconds. The second response time monitor bracket is from the first response time boundary up to and including the second response time boundary.
9	Third response time boundary (zoned (10,0)) in milliseconds. The third response time monitor bracket is from the second response time boundary up to and including the third response time boundary.
10	Fourth response time boundary (zoned (10,0)) in milliseconds. The fourth response time monitor bracket is from the third response time boundary up to and including the fourth response time boundary. Responses greater than the fourth response time boundary fall under the fifth response time monitor bracket.
11	System ASP capacity in KB (zone (10,0)). Total number of bytes of auxiliary storage allocated to the system ASP for the storage of data.
12	Checksum protection on (Y/N).
13	Number of configured CPUs (PD (3,0)).
C	Resource name of the communications IOP.
14	First remote response time boundary (zoned (10,0)) in milliseconds. The first response time monitor bracket is from 0 up to and including the first response time boundary.
15	Second remote response time boundary (zoned (10,0)) in milliseconds. The second response time monitor bracket is from the first response time boundary up to and including the second response time boundary.
16	Third remote response time boundary (zoned (10,0)) in milliseconds. The third response time monitor bracket is from the second response time boundary up to and including the third response time boundary.
17	Fourth remote response time boundary (zoned (10,0)) in milliseconds. The fourth response time monitor bracket is from the third response time boundary up to and including the fourth response time boundary. Responses greater than the fourth response time boundary fall under the fifth response time monitor bracket.
D	Resource name of the storage IOP.
E	Resource name of the device.
F	File level (PD(2,0)). Specifies the level of the Performance Monitor files. The value in this field is '1', and will be incremented each time the format of any of the Performance Monitor files changes.
I	Interval (PD(2,0)). The time interval (in minutes) between each collection of system performance data that was specified with the Start Performance Monitor command.
L	Resource name of the local work station controller.
N	Resource name of the line.
O	Resource name of the controller.
R	Version number (PD(2,0)), followed by release number (PD(3,1)).
S	System name (character 8).
T	Trace type (character 5). Specifies the type of internal trace that was started with the Start Performance Monitor command (*NONE or *ALL).

5. For GIOPA, GIOPB, GUNITA, and GUNITB, the value placed in the fields is -1. These entries do not apply to all records in QAPMCONF.

File Name: QAPMSYS

System Interval File Entries: The following terms are used in Figure 6-2, and are repeated for each group of jobs.

- Number of database read operations. Total number of physical read operations for database functions.

- Number of nondatabase read operations. Total number of physical read operations for nondatabase functions.
- Number of write operations. Total number of physical write operations.
- Number of print lines. Number of lines written by the program. This number does not reflect what is actually printed. Spooled files can be ended, or printed with multiple copies.
- Number of database writes/reads (logical). Number of times the database module was called. This number does not include I/O operations to readers/writers, or I/O operations caused by the Copy Spooled File (CPYSPLF) or Display Spooled File (DSPSPLF) command. If SEQONLY(*YES) is in effect, these numbers show each block of records read, not the number of individual records read.
- Number of communications writes/reads (logical). These do not include remote work station activity. They include only activity related to OS/400-ICF files when the I/O is for a communications device.
- Number of times PAG was brought. Total number of times the program access group (PAG) was brought into main storage after it was purged when running with PURGE(*YES) parameter.
- Number of times PAG was purged. Total number of times the PAG was purged when the job entered a long wait state and was running with PURGE(*YES) option.

Note: Blocked I/O is considered one database transaction.

Figure A-3 on page A-10 shows which system performance data is collected.

Figure A-3 (Page 1 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
SYDPGF	Directory page faults: Number of times a page of the auxiliary storage directory was transferred to main storage for a look-up or an allocation operation.	PD (11,0)	20
SYAPGF	Access group member page faults: Number of times a page of an object contained in an access group was transferred to main storage independently of the access group. This transfer occurs when the containing access group was purged, or because portions of the containing access group are displaced from main storage.	PD (11,0)	26
SYMPGF	Microcode page faults: Number of times a page of microcode was transferred to main storage.	PD (11,0)	32
SYMCTR	Microtask read operations: Number of transfers of one or more pages of data from auxiliary storage because of a microtask rather than a process.	PD (11,0)	38
SYMCTW	Microtask write operations: Number of transfers of one or more pages of data from main storage to auxiliary storage because of a microtask rather than a process.	PD (11,0)	44
SYSASP	System auxiliary storage pools space available: Number of bytes of space on auxiliary storage available for allocation in the system ASP that is not currently assigned to machine interface (MI) objects or internal machine functions.	PD (11,0)	50

Figure A-3 (Page 2 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SYPRMW	Permanent data transferred from main storage: Number of 512-byte blocks of permanent data transferred from main storage to the system ASP in auxiliary storage since the last sample.	PD (11,0)	56
SYXSRW	Redundancy data transferred from main storage: Number of 512-byte blocks of redundant data transferred from main storage to the system ASP in auxiliary storage since the last sample.	PD (11,0)	62
	Reserved.	C (18)	68
SYEAOT	Total number of effective address overflow exceptions.	PD (11,0)	86
SYEAOL	Total number of effective address-length overflow exceptions.	PD (11,0)	92
SYBSYC	Busy count: Total number of busy exceptions.	PD (11,0)	98
SYSIZC	Size count: Total number of size exceptions.	PD (11,0)	104
SYDECD	Decimal data count: Total number of decimal data exceptions.	PD (11,0)	110
SYSEZC	Seize count: Total number of seize wait exceptions.	PD (11,0)	116
SYSYNL	Synchronous lock conflict count.	PD (11,0)	122
SYASYL	Asynchronous lock conflict count.	PD (11,0)	128
SYVFYC	Verify count.	PD (11,0)	134
SYAUTH	Authority look-up count.	PD (11,0)	140
SYCHNB	Channel busy.	PD (11,0)	146
SYEXPN	Total number of exceptions.	PD (11,0)	152
	Reserved.	C (30)	158
SYLRT1	Transactions in first response time monitor bracket: Total number of local work station transactions with response time less than the value of boundary 1 specified on the STRPFRMON command.	PD (9,0)	188
SYLRT2	Transactions in second response time monitor bracket: Total number of local work station transactions with response time less than the value of boundary 2 and greater than the value of boundary 1 specified on the STRPFRMON command.	PD (9,0)	193
SYLRT3	Transactions in third response time monitor bracket: Total number of local work station transactions with response time less than the value of boundary 3 and greater than the value of boundary 2 specified on the STRPFRMON command.	PD (9,0)	198
SYLRT4	Transactions in fourth response time monitor bracket: Total number of local work station transactions with response time less than the value of boundary 4 and greater than the value of boundary 3 specified on the STRPFRMON command.	PD (9,0)	203
SYLRT5	Transactions in fifth response time monitor bracket: Total number of local work station transactions with response time greater than the value of boundary 5 specified on STRPFRMON command.	PD (9,0)	208
SDCPU	Total processing unit time used (in milliseconds) by target Distributed Data Management (DDM) job.	PD (11,0)	213
SDRES1	Reserved.	PD (11,0)	219
SDRES2	Reserved.	PD (11,0)	225
SDPRTL	Total number of print lines of all target DDM jobs.	PD (11,0)	231

Figure A-3 (Page 3 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SDPRTP	Total number of print pages of all target DDM jobs.	PD (11,0)	237
SDSPD	Total count of suspended time of target DDM jobs.	PD (11,0)	243
SDRRT	Total count of time a target DDM job waited during rerouting.	PD (11,0)	249
SDNEW	Number of new target DDM job.	PD (11,0)	255
SDTERM	Number of ended target DDM jobs.	PD (11,0)	261
SDJBCT	Number of DDM jobs.	PD (11,0)	267
SDPDBR	Total number of physical synchronous database reads by target DDM jobs.	PD (11,0)	273
SDPNDB	Total number of physical synchronous nondatabase reads by target DDM jobs.	PD (11,0)	279
SDPWRT	Total number of physical synchronous database and nondatabase writes by target DDM jobs.	PD (11,0)	285
SDLDBR	Total number of logical database reads by target DDM jobs.	PD (11,0)	291
SDLDBW	Total number of logical database writes by target DDM jobs.	PD (11,0)	297
SDLDBU	Total number of miscellaneous database operations by target DDM jobs.	PD (11,0)	303
SDCMPT	Total number of communications writes by target DDM jobs.	PD (11,0)	309
SDCMGT	Total number of communications reads by target DDM jobs.	PD (11,0)	315
SDBRG	Number of PURGE(*YES) transactions.	PD (11,0)	321
SDPRG	Number of DYNAMIC PURGE(*NO) transactions.	PD (11,0)	327
SDNDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions by target DDM jobs.	PD (11,0)	333
SDDBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions by target DDM jobs.	PD (11,0)	339
SDANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for nondatabase functions by target DDM jobs.	PD (11,0)	345
SDADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions by target DDM jobs.	PD (11,0)	351
SDANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for nondatabase functions by target DDM jobs.	PD (11,0)	357
SDADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions by target DDM jobs.	PD (11,0)	363
SDPW	Number of permanent writes by target DDM jobs.	PD (11,0)	369
SDCS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by target DDM jobs.	PD (11,0)	375

Figure A-3 (Page 4 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SDPAGF	Number of PAG faults. Total number of times the program access group (PAG) was referred to by target DDM jobs, but was not in main storage.	PD (11,0)	381
SDEAO	Number of effective address overflow exceptions by target DDM jobs.	PD (11,0)	387
SDOBIN	Number of binary overflows by target DDM jobs.	PD (11,0)	393
SDODEC	Number of decimal overflows by target DDM jobs.	PD (11,0)	399
SDOFLP	Number of floating point overflows by target DDM jobs.	PD (11,0)	405
SDIPF	Number of times a target distributed data management (DDM) job had a page fault on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	411
SDWIO	Number of times a target distributed data management (DDM) job explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	417
SWCPU	Total processing unit time (in milliseconds) used by PC Support applications.	PD (11,0)	423
SWRES1	Reserved.	PD (11,0)	429
SWRES2	Reserved.	PD (11,0)	435
SWPRTL	Total number of print lines of all PC Support application jobs.	PD (11,0)	441
SWPRTP	Total number of print pages of all PC Support application jobs.	PD (11,0)	447
SWSPD	Total time PC Support application jobs were suspended.	PD (11,0)	453
SWRRT	Total time a PC Support applications job waited during rerouting.	PD (11,0)	459
SWNEW	Number of started PC Support application jobs.	PD (11,0)	465
SWTERM	Number of ended PC Support application jobs.	PD (11,0)	471
SWJBCT	Number of PC support jobs.	PD (11,0)	477
SWPDBR	Total number of physical synchronous database reads by PC Support application jobs.	PD (11,0)	483
SWPNDB	Total number of physical synchronous nondatabase reads by PC Support application jobs.	PD (11,0)	489
SWPWRT	Total number of physical synchronous database and nondatabase writes by PC Support application jobs.	PD (11,0)	495
SWLDBR	Total number of logical database reads by PC Support application jobs.	PD (11,0)	501
SWLDBW	Total number of logical database writes by PC Support application jobs.	PD (11,0)	507
SWLDBU	Total number of miscellaneous database operations by PC Support application jobs.	PD (11,0)	513
SWCMPT	Total number of communications writes by PC Support application jobs.	PD (11,0)	519
SWCMGT	Total number of communications reads by PC Support application jobs.	PD (11,0)	525
SWBRG	Number of PURGE(*YES) transactions.	PD (11,0)	531
SWPRG	Number of DYNAMIC PURGE(*NO) transactions.	PD (11,0)	537
SWNDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions by PC Support applications.	PD (11,0)	543

Figure A-3 (Page 5 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SWDBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions by PC Support applications.	PD (11,0)	549
SWANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for nondatabase functions by PC Support applications.	PD (11,0)	555
SWADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions by PC Support applications.	PD (11,0)	561
SWANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for nondatabase functions by PC Support applications.	PD (11,0)	567
SWADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions by PC Support applications.	PD (11,0)	573
SWPW	Number of permanent writes by PC Support applications.	PD (11,0)	579
SWCS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by this job by PC Support applications.	PD (11,0)	585
SWPAGF	Number of PAG faults. Total number of times the program access group (PAG) was referred to by PC Support applications, but was not in main storage.	PD (11,0)	591
SWEAO	Number of effective address overflow exceptions by PC Support applications.	PD (11,0)	597
SWOBIN	Number of binary overflows by PC Support applications.	PD (11,0)	603
SWODEC	Number of decimal overflows by PC Support applications.	PD (11,0)	609
SWOFLP	Number of floating point overflows by PC Support applications.	PD (11,0)	615
SWIPF	Number of times a PC Support application job had a page fault on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	621
SWWIO	Number of times a PC Support application job explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	627
SPCPU	Total processing unit time (in milliseconds) used by pass-through target jobs.	PD (11,0)	633
SPRES1	Total transaction time by pass-through target jobs.	PD (11,0)	639
SPRES2	Total number of transactions by pass-through target jobs.	PD (11,0)	645
SPPRTL	Total number of print lines of all pass-through target jobs.	PD (11,0)	651
SPPRTP	Total number of print pages of all pass-through target jobs.	PD (11,0)	657
SPSPD	Total count of suspended time of pass-through target jobs.	PD (11,0)	663
SPRRT	Total count of time a pass-through target job waited during rerouting.	PD (11,0)	669
SPNEW	Number of started pass-through target jobs.	PD (11,0)	675
SPTERM	Number of ended pass-through target jobs.	PD (11,0)	681
SPJBCT	Number of pass-through jobs.	PD (11,0)	687
SPPDBR	Total number of physical synchronous database reads by pass-through target jobs.	PD (11,0)	693

Figure A-3 (Page 6 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SPPNDB	Total number of physical synchronous nondatabase reads by pass-through target jobs.	PD (11,0)	699
SPPWRT	Total number of physical synchronous database and nondatabase writes by pass-through target jobs.	PD (11,0)	705
SPLDBR	Total number of logical database reads by pass-through target jobs.	PD (11,0)	711
SPLDBW	Total number of logical database writes by pass-through target jobs.	PD (11,0)	717
SPLDBU	Total number of miscellaneous database operations by pass-through target jobs.	PD (11,0)	723
SPCMPT	Total number of communications writes by pass-through target jobs.	PD (11,0)	729
SPCMGT	Total number of communications reads by pass-through target jobs.	PD (11,0)	735
SPBRG	Number of PURGE(*YES) transactions.	PD (11,0)	741
SPPRG	Number of DYNAMIC PURGE(*NO) transactions.	PD (11,0)	747
SPNDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions by pass-through target jobs.	PD (11,0)	753
SPDBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions by pass-through target jobs.	PD (11,0)	759
SPANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for nondatabase functions by pass-through target jobs.	PD (11,0)	765
SPADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions by pass-through target jobs.	PD (11,0)	771
SPANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for nondatabase functions by pass-through target jobs.	PD (11,0)	777
SPADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions by pass-through target jobs.	PD (11,0)	783
SPPW	Number of permanent writes by pass-through target jobs.	PD (11,0)	789
SPCS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by this job by pass-through target jobs.	PD (11,0)	795
SPPAGF	Number of PAG faults: Total number of times the program access group (PAG) was referred to by pass-through target jobs, but was not in main storage.	PD (11,0)	801
SPEAO	Number of effective address overflow exceptions by pass-through target jobs.	PD (11,0)	807
SPOBIN	Number of binary overflows by pass-through target jobs.	PD (11,0)	813
SPODEC	Number of decimal overflows by pass-through target jobs.	PD (11,0)	819
SPOFLP	Number of floating point overflows by pass-through target jobs.	PD (11,0)	825
SPIPF	Number of times a pass-through target job had a page fault on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	831

Figure A-3 (Page 7 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SPWIO	Number of times a pass-through target job explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	837
SMCPU	Total processing unit time (in milliseconds) used by multiple requester terminal (MRT) jobs (System/36 environment only).	PD (11,0)	843
SMRES1	Reserved.	PD (11,0)	849
SMRES2	Reserved.	PD (11,0)	855
SMPRTL	Total number of print lines of all MRT jobs (System/36 environment only).	PD (11,0)	861
SMPRTP	Total number of print pages of all MRT jobs (System/36 environment only).	PD (11,0)	867
SMSPD	Total time MRT jobs (System/36 environment only) were suspended.	PD (11,0)	873
SMRRT	Total time a MRT job (System/36 environment only) waited during rerouting.	PD (11,0)	879
SMNEW	Number of started MRT jobs (System/36 environment only).	PD (11,0)	885
SMTERM	Number of ended MRT jobs (System/36 environment only).	PD (11,0)	891
SMJBCT	Number of MRT jobs (System/36 environment only).	PD (11,0)	897
SMPDBR	Total number of physical synchronous database reads by MRT jobs (System/36 environment only).	PD (11,0)	903
SMPNDB	Total number of physical synchronous nondatabase reads by MRT jobs (System/36 environment only).	PD (11,0)	909
SMPWRT	Total number of physical synchronous database and nondatabase writes by MRT jobs (System/36 environment only).	PD (11,0)	915
SMLDBR	Total number of logical database reads by MRT jobs (System/36 environment only).	PD (11,0)	921
SMLDBW	Total number of logical database writes by MRT jobs (System/36 environment only).	PD (11,0)	927
SMLDBU	Total number of miscellaneous database operations by MRT jobs (System/36 environment only).	PD (11,0)	933
SMCMPT	Total number of communications writes by MRT jobs (System/36 environment only).	PD (11,0)	939
SMCMGT	Total number of communications reads by MRT jobs (System/36 environment only).	PD (11,0)	945
SMBRG	Number of PURGE(*YES) transactions.	PD (11,0)	951
SMPRG	Number of DYNAMIC PURGE(*NO) transactions.	PD (11,0)	957
SMNDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions by MRT jobs (System/36 environment only).	PD (11,0)	963
SMDBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions by MRT jobs (System/36 environment only).	PD (11,0)	969
SMANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for nondatabase functions by MRT jobs (System/36 environment only).	PD (11,0)	975

Figure A-3 (Page 8 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SMADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions by MRT jobs (System/36 environment only).	PD (11,0)	981
SMANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for nondatabase functions by MRT jobs (System/36 environment only).	PD (11,0)	987
SMADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions by MRT jobs (System/36 environment only).	PD (11,0)	993
SMPW	Number of permanent writes by MRT jobs (System/36 environment only).	PD (11,0)	999
SMCS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by this job by MRT jobs (System/36 environment only).	PD (11,0)	1005
SMPAGF	Number of PAG faults: Total number of times the program access group (PAG) was referred to by MRT jobs (System/36 environment only), but was not in main storage.	PD (11,0)	1011
SMEAO	Number of effective address overflow exceptions by MRT jobs (System/36 environment only).	PD (11,0)	1017
SMOBIN	Number of binary overflows by MRT jobs (System/36 environment only).	PD (11,0)	1023
SMODEC	Number of decimal overflows by MRT jobs (System/36 environment only).	PD (11,0)	1029
SMOFLP	Number of floating point overflows by MRT jobs (System/36 environment only).	PD (11,0)	1035
SMIPF	Number of times a MRT job (System/36 environment only) had a page fault on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	1041
SMWIO	Number of times a MRT job (System/36 environment only) explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	1047
S6CPU	Total processing unit time (in milliseconds) used by System/36 environment jobs.	PD (11,0)	1053
S6TRNT	Total response time.	PD (11,0)	1059
S6TRNS	Number of transactions.	PD (11,0)	1065
S6PRTL	Total number of print lines of all System/36 environment jobs.	PD (11,0)	1071
S6PRTP	Total number of print pages of all System/36 environment jobs.	PD (11,0)	1077
S6SPD	Total time System/36 environment jobs were suspended.	PD (11,0)	1083
S6RRT	Total time a System/36 environment job waited during rerouting.	PD (11,0)	1089
S6NEW	Number of started System/36 environment jobs.	PD (11,0)	1095
S6TERM	Number of ended System/36 environment jobs.	PD (11,0)	1101
S6JBCT	Number of System/36 environment jobs.	PD (11,0)	1107
S6PDBR	Total number of physical synchronous database reads by System/36 environment jobs.	PD (11,0)	1113

Figure A-3 (Page 9 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
S6PNDB	Total number of physical synchronous nondatabase reads by System/36 environment jobs.	PD (11,0)	1119
S6PWRT	Total number of physical synchronous database and nondatabase writes by System/36 environment jobs.	PD (11,0)	1125
S6LDBR	Total number of logical database reads by System/36 environment jobs.	PD (11,0)	1131
S6LDBW	Total number of logical database writes by System/36 environment jobs.	PD (11,0)	1137
S6LDBU	Total number of miscellaneous database operations by System/36 environment jobs.	PD (11,0)	1143
S6CMPT	Total number of communications writes by System/36 environment jobs.	PD (11,0)	1149
S6CMGT	Total number of communications reads by System/36 environment jobs.	PD (11,0)	1155
S6BRG	Number of PURGE(*YES) transactions.	PD (11,0)	1161
S6PRG	Number of DYNAMIC PURGE(*NO) transactions.	PD (11,0)	1167
S6NDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions by System/36 environment jobs.	PD (11,0)	1173
S6DBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions by System/36 environment jobs.	PD (11,0)	1179
S6ANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for nondatabase functions by System/36 environment jobs.	PD (11,0)	1185
S6ADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions by System/36 environment jobs.	PD (11,0)	1191
S6ANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for nondatabase functions by System/36 environment jobs.	PD (11,0)	1197
S6ADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions by System/36 environment jobs.	PD (11,0)	1203
S6PW	Number of permanent writes by System/36 environment jobs.	PD (11,0)	1209
S6CS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by this job by System/36 environment jobs.	PD (11,0)	1215
S6PAGF	Number of PAG faults: Total number of times the program access group (PAG) was referred to by System/36 environment jobs, but was not in main storage.	PD (11,0)	1221
S6EAO	Number of effective address overflow exceptions by System/36 environment jobs.	PD (11,0)	1227
S6OBIN	Number of binary overflows by System/36 environment jobs.	PD (11,0)	1233
S6ODEC	Number of decimal overflows by System/36 environment jobs.	PD (11,0)	1239

Figure A-3 (Page 10 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
S6OFLP	Number of floating point overflows by System/36 environment jobs.	PD (11,0)	1245
S6IPF	Number of times a System/36 environment job had a page fault on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	1251
S6WIO	Number of times a System/36 environment job explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	1257
SECPU	Total processing unit time (in milliseconds) used by communications batch jobs.	PD (11,0)	1263
SERES1	Reserved.	PD (11,0)	1269
SERES2	Reserved.	PD (11,0)	1275
SEPRTL	Total number of print lines of all communications batch jobs.	PD (11,0)	1281
SEPRTP	Total number of print pages of all communications batch jobs.	PD (11,0)	1287
SESPD	Total time communications batch jobs were suspended.	PD (11,0)	1293
SERRT	Total time a communications batch job waited during rerouting.	PD (11,0)	1299
SENEW	Number of started communications batch jobs.	PD (11,0)	1305
SETERM	Number of ended communications batch jobs.	PD (11,0)	1311
SEJBCT	Number of communications batch jobs.	PD (11,0)	1317
SEPDBR	Total number of physical synchronous database reads by communications batch jobs.	PD (11,0)	1323
SEPNDB	Total number of physical synchronous nondatabase reads by communications batch jobs.	PD (11,0)	1329
SEPWRT	Total number of physical synchronous database and nondatabase writes by communications batch jobs.	PD (11,0)	1335
SELDBR	Total number of logical database reads by communications batch jobs.	PD (11,0)	1341
SELDBW	Total number of logical database writes by communications batch jobs.	PD (11,0)	1347
SELDBU	Total number of miscellaneous database operations by communications batch jobs.	PD (11,0)	1353
SECMPT	Total number of communications writes by communications batch jobs.	PD (11,0)	1359
SECMGT	Total number of communications reads by communications batch jobs.	PD (11,0)	1365
SEBRG	Number of PURGE(*YES) transactions.	PD (11,0)	1371
SEPRG	Number of DYNAMIC PURGE(*NO) transactions.	PD (11,0)	1377
SENDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions by communications batch jobs.	PD (11,0)	1383
SEDBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions by communications batch jobs.	PD (11,0)	1389
SEANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for nondatabase functions by communications batch jobs.	PD (11,0)	1395
SEADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions by communications batch jobs.	PD (11,0)	1401

Figure A-3 (Page 11 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SEANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for nondatabase functions by communications batch jobs.	PD (11,0)	1407
SEADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions by communications batch jobs.	PD (11,0)	1413
SEPW	Number of permanent writes by communications batch jobs.	PD (11,0)	1419
SECS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by communications batch jobs.	PD (11,0)	1425
SEPAGF	Number of PAG faults: Total number of times the program access group (PAG) was referred to by communications batch jobs, but was not in main storage.	PD (11,0)	1431
SEEAO	Number of effective address overflow exceptions by communications batch jobs.	PD (11,0)	1437
SEOBIN	Number of binary overflows by communications batch jobs.	PD (11,0)	1443
SEODEC	Number of decimal overflows by communications batch jobs.	PD (11,0)	1449
SEOFLP	Number of floating point overflows by communications batch jobs.	PD (11,0)	1455
SEIPF	Number of times a communications batch job had a page fault on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	1461
SEWIO	Number of times a communications batch job explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	1467
SACPU	Total processing unit time (in milliseconds) used by autostart jobs.	PD (11,0)	1473
SARES1	Reserved.	PD (11,0)	1479
SARES2	Reserved.	PD (11,0)	1485
SAPRTL	Total number of print lines of all autostart jobs.	PD (11,0)	1491
SAPRTP	Total number of print pages of all autostart jobs.	PD (11,0)	1497
SASPD	Total time autostart jobs were suspended.	PD (11,0)	1503
SARRT	Total time an autostart job waited during rerouting.	PD (11,0)	1509
SANEW	Number of started autostart jobs.	PD (11,0)	1515
SATERM	Number of ended autostart jobs.	PD (11,0)	1521
SAJBCT	Number of autostart jobs.	PD (11,0)	1527
SAPDBR	Total number of physical synchronous database reads by autostart jobs.	PD (11,0)	1533
SAPNDB	Total number of physical synchronous nondatabase reads by autostart jobs.	PD (11,0)	1539
SAPWRT	Total number of physical synchronous database and nondatabase writes by autostart jobs.	PD (11,0)	1545
SALDBR	Total number of logical database reads by autostart jobs.	PD (11,0)	1551
SALDBW	Total number of logical database writes by autostart jobs.	PD (11,0)	1557
SALDBU	Total number of miscellaneous database operations by autostart jobs.	PD (11,0)	1563
SACMPT	Total number of communications writes by autostart jobs.	PD (11,0)	1569

Figure A-3 (Page 12 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SACMGT	Total number of communications reads by autostart jobs.	PD (11,0)	1575
SABRG	Number of PURGE(*YES) transactions.	PD (11,0)	1581
SAPRG	Number of DYNAMIC PURGE(*NO) transactions.	PD (11,0)	1587
SANDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions by communications batch jobs.	PD (11,0)	1593
SADBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions by autostart jobs.	PD (11,0)	1599
SAANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for nondatabase functions by autostart jobs.	PD (11,0)	1605
SAADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions by autostart jobs.	PD (11,0)	1611
SAANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for nondatabase functions by autostart jobs.	PD (11,0)	1617
SAADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions by autostart jobs.	PD (11,0)	1623
SAPW	Number of permanent writes by autostart jobs.	PD (11,0)	1629
SACS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by autostart jobs.	PD (11,0)	1635
SAPAGF	Number of PAG faults: Total number of times the program access group (PAG) was referred to by autostart jobs, but was not in main storage.	PD (11,0)	1641
SAEAO	Number of effective address overflow exceptions by autostart jobs.	PD (11,0)	1647
SAOBIN	Number of binary overflows by autostart jobs.	PD (11,0)	1653
SAODEC	Number of decimal overflows by autostart jobs.	PD (11,0)	1659
SAOFLP	Number of floating point overflows by autostart jobs.	PD (11,0)	1665
SAIPF	Number of times an autostart job had a page fault on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	1671
SAWIO	Number of times an autostart job explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	1677
SBCPU	Total processing unit time (in milliseconds) used by batch jobs.	PD (11,0)	1683
SBRES1	Reserved.	PD (11,0)	1689
SBRES2	Reserved.	PD (11,0)	1695
SBPRTL	Total number of print lines of all batch jobs.	PD (11,0)	1701
SBPRTP	Total number of print pages of all batch jobs.	PD (11,0)	1707
SBSPD	Total time batch jobs were suspended.	PD (11,0)	1713
SBRRT	Total time a batch job waited during rerouting.	PD (11,0)	1719
SBNEW	Number of started batch jobs.	PD (11,0)	1725

Figure A-3 (Page 13 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SBTERM	Number of ended batch jobs.	PD (11,0)	1731
SBJBCT	Number of batch jobs.	PD (11,0)	1737
SBPDBR	Total number of physical synchronous database reads by batch jobs.	PD (11,0)	1743
SBPNDB	Total number of physical synchronous nondatabase reads by batch jobs.	PD (11,0)	1749
SBPWRT	Total number of physical synchronous database and nondatabase writes by batch jobs.	PD (11,0)	1755
SBLDBR	Total number of logical database reads by batch jobs.	PD (11,0)	1761
SBLDBW	Total number of logical database writes by batch jobs.	PD (11,0)	1767
SBLDBU	Total number of miscellaneous database operations by batch jobs.	PD (11,0)	1773
SBCMPT	Total number of communications writes by batch jobs.	PD (11,0)	1779
SBCMGT	Total number of communications reads by batch jobs.	PD (11,0)	1785
SBBRG	Number of PURGE(*YES) transactions.	PD (11,0)	1791
SBPRG	Number of DYNAMIC PURGE(*NO) transactions.	PD (11,0)	1797
SBNDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions by batch jobs.	PD (11,0)	1803
SBDBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions by batch jobs.	PD (11,0)	1809
SBANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for nondatabase functions by batch jobs.	PD (11,0)	1815
SBADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions by batch jobs.	PD (11,0)	1821
SBANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for database functions by batch jobs.	PD (11,0)	1827
SBADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions by batch jobs.	PD (11,0)	1833
SBPW	Number of permanent writes by batch jobs.	PD (11,0)	1839
SBCS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by batch jobs.	PD (11,0)	1845
SBPAGF	Number of PAG faults: Total number of times the program access group (PAG) was referred to by batch jobs, but was not in main storage.	PD (11,0)	1851
SBEAO	Number of effective address overflow exceptions by batch jobs.	PD (11,0)	1857
SBOBIN	Number of binary overflows by batch jobs.	PD (11,0)	1863
SBODEC	Number of decimal overflows by batch jobs.	PD (11,0)	1869
SBOFLP	Number of floating point overflows by batch jobs.	PD (11,0)	1875

Figure A-3 (Page 14 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SBIPF	Number of times a batch job had a page fault on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	1881
SBWIO	Number of times a batch job explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	1887
SICPU	Total processing unit time (in milliseconds) used by interactive jobs.	PD (11,0)	1893
SITRNT	Total transaction time by interactive jobs.	PD (11,0)	1899
SITRNS	Total number of transactions by interactive jobs.	PD (11,0)	1905
SIPRTL	Total number of print lines of all interactive jobs.	PD (11,0)	1911
SIPRTP	Total number of print pages of all interactive jobs.	PD (11,0)	1917
SISPD	Total time interactive jobs were suspended.	PD (11,0)	1923
SIRRT	Total time an interactive job waited during rerouting.	PD (11,0)	1929
SINEW	Number of started interactive jobs.	PD (11,0)	1935
SITERM	Number of ended interactive jobs.	PD (11,0)	1941
SIJBCT	Number of ended interactive jobs.	PD (11,0)	1947
SIPDBR	Total number of physical synchronous database reads by interactive jobs.	PD (11,0)	1953
SIPNDB	Total number of physical synchronous nondatabase reads by interactive jobs.	PD (11,0)	1959
SIPWRT	Total number of physical synchronous database and nondatabase writes by interactive jobs.	PD (11,0)	1965
SILDBR	Total number of logical database reads by interactive jobs.	PD (11,0)	1971
SILDBW	Total number of logical database writes by interactive jobs.	PD (11,0)	1977
SILDBU	Total number of miscellaneous database operations by interactive jobs.	PD (11,0)	1983
SICMPT	Total number of communications writes by interactive jobs.	PD (11,0)	1989
SICMGT	Total number of communications reads by interactive jobs.	PD (11,0)	1995
SIBRG	Number of PURGE(*YES) transactions.	PD (11,0)	2001
SIPRG	Number of DYNAMIC PURGE(*NO) transactions.	PD (11,0)	2007
SINDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions by interactive jobs.	PD (11,0)	2013
SIDBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions by interactive jobs.	PD (11,0)	2019
SIANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for nondatabase functions by interactive jobs.	PD (11,0)	2025
SIADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions by interactive jobs.	PD (11,0)	2031
SIANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for nondatabase functions by interactive jobs.	PD (11,0)	2037

Figure A-3 (Page 15 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SIADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions by interactive jobs.	PD (11,0)	2043
SIPW	Number of permanent writes by interactive jobs.	PD (11,0)	2049
SICS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by this job by interactive jobs.	PD (11,0)	2055
SIPAGF	Number of PAG faults: Total number of times the program access group (PAG) was referred to by interactive jobs but was not in main storage.	PD (11,0)	2061
SIEAO	Number of effective address overflow exceptions by interactive jobs.	PD (11,0)	2067
SIOBIN	Number of binary overflows by interactive jobs.	PD (11,0)	2073
SIODEC	Number of decimal overflows interactive jobs.	PD (11,0)	2079
SIOFLP	Number of floating point overflows by interactive jobs.	PD (11,0)	2085
SIIPF	Number of times an interactive job had a page fault on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	2091
SIWIO	Number of times an interactive job explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	2097
SXCPU	Total processing unit time (in milliseconds) used by start CPF (SCPF) jobs/spool reader/spool writer.	PD (11,0)	2103
SXRES1	Reserved.	PD (11,0)	2109
SXRES2	Reserved.	PD (11,0)	2115
SXPRTL	Total number of print lines of all SCPF jobs/spool reader/spool writer.	PD (11,0)	2121
SXPRTP	Total number of print pages of all SCPF jobs/spool reader/spool writer.	PD (11,0)	2127
SXSPD	Total time SCPF jobs/spool reader/spool writer were suspended.	PD (11,0)	2133
SXRRT	Total time SCPF jobs or spool reader/writer waited during rerouting.	PD (11,0)	2139
SXNEW	Number of started SCPF jobs or spool reader/writer.	PD (11,0)	2145
SXTERM	Number of ended SCPF jobs or spool reader/writer.	PD (11,0)	2151
SXJBCT	Number of SCPF jobs or spool reader/writer.	PD (11,0)	2157
SXPDBR	Total number of physical synchronous database reads by SCPF jobs/spool reader/spool writer.	PD (11,0)	2163
SXPNDB	Total number of physical synchronous nondatabase reads by SCPF jobs/spool reader/spool writer.	PD (11,0)	2169
SXPWRT	Total number of physical synchronous database and nondatabase writes by SCPF jobs/spool reader/spool writer.	PD (11,0)	2175
SXLDBR	Total number of logical database reads by SCPF jobs/spool reader/spool writer.	PD (11,0)	2181
SXLDBW	Total number of logical database writes by SCPF jobs/spool reader/spool writer.	PD (11,0)	2187
SXLDBU	Total number of miscellaneous database operations by SCPF jobs/spool reader/spool writer.	PD (11,0)	2193
SXCMPT	Total number of communications writes by SCPF jobs/spool reader/spool writer.	PD (11,0)	2199

Figure A-3 (Page 16 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SXCMGT	Total number of communications reads by SCPF jobs/spool reader/spool writer.	PD (11,0)	2205
SXBRG	Number of PURGE(*YES) transactions.	PD (11,0)	2211
SXPRG	Number of DYNAMIC PURGE(*NO) transactions.	PD (11,0)	2217
SXNDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions by SCPF jobs/spool reader/spool writer.	PD (11,0)	2223
SXDBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions by SCPF jobs/spool reader/spool writer.	PD (11,0)	2229
SXANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for database functions by SCPF jobs/spool reader/spool writer.	PD (11,0)	2235
SXADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions by SCPF jobs/spool reader/spool writer.	PD (11,0)	2241
SXANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for nondatabase functions by SCPF jobs/spool reader/spool writer.	PD (11,0)	2247
SXADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions by SCPF jobs/spool reader/spool writer.	PD (11,0)	2253
SXPW	Number of permanent writes by SCPF jobs/spool reader/spool writer.	PD (11,0)	2259
SXCS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by SCPF jobs/spool reader/spool writer.	PD (11,0)	2265
SXPAGF	Number of PAG faults: Total number of times the program access group (PAG) was referred to by SCPF jobs/spool reader/spool writer, but was not in main storage	PD (11,0)	2271
SXEAO	Number of effective address overflow exceptions by SCPF jobs/spool reader/spool writer.	PD (11,0)	2277
SXOBIN	Number of binary overflows by SCPF jobs/spool reader/spool writer.	PD (11,0)	2283
SXODEC	Number of decimal overflows by SCPF jobs/spool reader/spool writer.	PD (11,0)	2289
SXOFLP	Number of floating point overflows by SCPF jobs/spool reader/spool writer.	PD (11,0)	2295
SXIPF	Number of times a SCPF job or spool reader or spool writer job had a page fault on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	2301
SXWIO	Number of times a SCPF job or spool reader or spool writer job explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	2307
SHCPU	Total processing unit time (in milliseconds) used by microcode/system jobs.	PD (11,0)	2313
SMPLP	Machine pool paging: Number of pages transferred in and out of machine pool.	PD (11,0)	2319

Figure A-3 (Page 17 of 17). System Performance Data (Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
SMUPL	Highest user pool paging: Number of pages transferred in and out of user pool.	PD (11,0)	2325
SUPLI	Pool with highest paging: Pool number with highest number of pages transferred in and out.	C (2)	2331
SMXDU	Maximum disk utilization.	PD (11,0)	2333
SMXDUI	Actuator with maximum utilization.	C (4)	2339
SMMMT	Time (in seconds) spent at MRTMAX by all MRT requests.	PD (11,0)	2343
SMME	Number of requesters that routed to a MRT.	PD (11,0)	2349
SYSCPU	Total processing time (in milliseconds) used by the first (or only) processing unit.	PD (9,0)	2355
SYCPU2	Total processing time (in milliseconds) used by the second processing unit. This value is zero if there is only one processing unit.	PD (9,0)	2360
SYCPU3	Total processing time (in milliseconds) used by the third processing unit. This value is zero if there are two or fewer processing units.	PD (9,0)	2365
SYCPU4	Total processing time (in milliseconds) used by the fourth processing unit. This value is zero if there are three or fewer processing units.	PD (9,0)	2370

File Name: QAPMJOBS

JC/JI Database File Entries: Figure A-4 lists the various fields in the job completion (JC) and job interval (JI) file.

Note: In Figure A-4, *job* means job or task.

Figure A-4 (Page 1 of 5). Job Data (One Entry for Each Job in the System, Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) for job interval entry and job completion date, and time (hh:mm:ss) for job completion entry.	C (12)	4
INTSEC	Elapsed interval seconds.	PD (7,0)	16
JBSSYS	Name of the subsystem the job is running in.	C (10)	20
JBSLIB	Name of the library the subsystem is in.	C (10)	30
JBNAME	Job name/work station name.	C (10)	40
JBUSER	Job user.	C (10)	50
JBNBR	Job number.	C (6)	60
JBACCO	Job accounting code. Field cannot be displayed.	C (15)	66
JBTYPE	Job type (A,B,H,I,M,R,S,V,W,X) (see Note 1 on page A-30).	C (1)	81
JBSTYP	Job subtype (see Note 2 on page A-30).	C (1)	82
JBFLAG	Job flag (see Note 3 on page A-30). Field cannot be displayed.	C (2)	83
JBS36E	Is job running in System/36 environment? (Y/N)	C (1)	85
	Reserved.	C (1)	86

Figure A-4 (Page 2 of 5). Job Data (One Entry for Each Job in the System, Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
JBPOOL	Job pool.	C (2)	87
JBPRTY	Job priority.	C (3)	89
JBCPU	Processing unit time (in milliseconds) used.	PD (11,0)	92
JBRSP	Total transaction time (in seconds): Field has a value other than zero only if this is an interactive job or a pass-through target job.	PD (11,0)	98
JBSLC	Time-slice value (in milliseconds).	PD (11,0)	104
JBNTR	Number of transactions (5250 only): Field has a value other than zero only if this is an interactive job or a pass-through target job.	PD (11,0)	110
JBDBR	Number of synchronous database reads: Total number of physical synchronous database read operations for database functions.	PD (11,0)	116
JBNDB	Number of synchronous nondatabase reads: Total number of physical synchronous nondatabase read operations for nondatabase functions.	PD (11,0)	122
JBWRT	Number of writes: Total number of physical database and nondatabase write operations.	PD (11,0)	128
JBAW	Total number of transitions from active state to wait state for this job.	PD (11,0)	134
JBWI	Total number of transitions from wait state to ineligible state for this job.	PD (11,0)	140
JBAI	Total number of transitions from active state to ineligible state for this job.	PD (11,0)	146
JBPLN	Number of print lines: Number of lines written by the program. This does not reflect what is actually printed. Spooled files can be ended, or printed with multiple copies.	PD (11,0)	152
JBPPG	Number of print pages.	PD (11,0)	158
JBPFL	Number of print files.	PD (11,0)	164
JBLWT	Number of database writes (logical): Number of times the internal database write function was called. This does not include I/O operations to readers/writers, or I/O operations caused by the CPYSPLF or DSPSPLF command. If SEQONLY(*YES) is specified, these numbers show each block of records read, not the number of individual records read.	PD (11,0)	170
JBLRD	Number of database reads (logical): Number of times the database module was called. This does not include I/O operations to readers/writers, or I/O operations caused by the CPYSPLF or DSPSPLF command. If SEQONLY(*YES) is specified, these numbers show each block of records read, not the number of individual records read.	PD (11,0)	176
JBDBU	Number of miscellaneous database operations: Updates, deletes, force-end-of-data, and releases (logical).	PD (11,0)	182
JBCPT	Number of communications writes: These do not include remote work station activity. They include only activity related to OS/400-ICF files when the I/O is for an OS/400-ICF device.	PD (11,0)	188
JBCGT	Number of communications reads (logical): These do not include remote work station activity. They include only activity related to OS/400-ICF files when the I/O is for an OS/400-ICF device.	PD (11,0)	194
JBSPD	Total suspended time (in milliseconds).	PD (11,0)	200

Figure A-4 (Page 3 of 5). Job Data (One Entry for Each Job in the System, Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
JBRRT	Total time job waited during reroutes (in milliseconds).	PD (11,0)	206
JBLND	Line description: Name of the communications line this work station and its controller is attached to. This is only available for remote work stations.	C (10)	212
JBCUD	Controller description: Name of the controller this work station is attached to.	C (10)	222
JB2LND	Secondary line description (pass-through and emulation only).	C (10)	232
JB2CUD	Secondary controller description (pass-through and emulation only).	C (10)	242
JBBRG	Number of times PAG was brought in: Total number of times the program access group (PAG) was brought into main storage after it was purged when running with PURGE(*YES) parameter specified.	PD (9,0)	252
JBPRG	Number of times PAG was purged: Total number of times the program access group (PAG) was purged when the job entered a long wait and was running with PURGE(*YES) parameter specified.	PD (9,0)	257
JBNDW	Number of synchronous nondatabase writes: Total number of synchronous physical nondatabase write operations for nondatabase functions.	PD (11,0)	262
JBDBW	Number of synchronous database writes: Total number of synchronous physical database write operations for database functions.	PD (11,0)	268
JBANDW	Number of asynchronous nondatabase writes: Total number of asynchronous physical nondatabase write operations for nondatabase functions.	PD (11,0)	274
JBADBW	Number of asynchronous database writes: Total number of asynchronous physical database write operations for database functions.	PD (11,0)	280
JBANDR	Number of asynchronous nondatabase reads: Total number of asynchronous physical nondatabase read operations for nondatabase functions.	PD (11,0)	286
JBADBR	Number of asynchronous database reads: Total number of asynchronous physical database read operations for database functions.	PD (11,0)	292
JBPW	Number of synchronous permanent writes.	PD (11,0)	298
JBCS	Checksum I/Os. Total number of I/Os (reads and writes) performed for checksum updating due to protected writes caused by this job.	PD (11,0)	304
JBPAGF	Number of PAG faults. Total number of times the program access group (PAG) was referred to, but was not in main storage.	PD (11,0)	310
JBEAO	Number of effective address overflow exceptions.	PD (11,0)	316
JBOBIN	Number of binary overflows.	PD (11,0)	322
JBODEC	Number of decimal overflows.	PD (11,0)	328
JBOFLP	Number of floating point overflows.	PD (11,0)	334
JBIPF	Number of times a page fault occurred on an address that was currently part of an auxiliary storage I/O operation.	PD (11,0)	340
JBWIO	Number of times the process explicitly waited for outstanding asynchronous I/O operations to complete.	PD (11,0)	346
	Reserved.	PD (15,3)	352
	Reserved.	PD (15,3)	360
JIOPB	IOP bus number.	PD (3,0)	368

Figure A-4 (Page 4 of 5). Job Data (One Entry for Each Job in the System, Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
JIOPA	IOP bus address.	PD (3,0)	370
JBPORT	Work station port number.	PD (3,0)	372
JBSTN	Work station number.	PD (3,0)	374
JBPTSF	Pass-through source flag.	PD (1,0)	376
JBPTTF	Pass-through target flag.	PD (1,0)	377
JBEAF	Emulation active flag.	PD (1,0)	378
JBPCSF	PC Support application flag.	PD (1,0)	379
JBDDMF	Target DDM job flag.	PD (1,0)	380
JBMRTF	MRT flag.	PD (1,0)	381
JBROUT	The routing entry index for the subsystem this job is in.	PD (5,0)	382
JBIOA	IOA number.	PD (3,0)	385
JBWSID	Work station address: The address of the work station on which the job is running. Character data cannot be displayed.	C (5)	387
JBAPT	This field is for IBM internal use only.	PD (11,0)	392
JBNSW	This field is for IBM internal use only.	PD (11,0)	398
JBSST	This field is for IBM internal use only.	PD (11,0)	404
JBQT2	This field is for IBM internal use only.	PD (11,0)	410
JBCDR	This field is for IBM internal use only.	PD (11,0)	416
JBCDS	This field is for IBM internal use only.	PD (11,0)	422
JBAIQT	Total application queuing time (in hundredths of seconds).	PD (11,0)	428
JBNAIQ	Number of application queuing transactions.	PD (11,0)	434
JBRUT	Total resource usage time (in seconds).	PD (11,0)	440
JBNRU	Number of resource usage transactions.	PD (11,0)	446
JBQT	Total queuing time to enter the MRT (in hundredths of seconds).	PD (11,0)	452
JBMMT	Total time spent at MRTMAX (in seconds).	PD (11,0)	458
JBNETQ	Total number of entries into the MRT.	PD (11,0)	464
JBPUTN	The number of times ACPUT was called to send user or control data. Calls that result in no data being sent are not counted.	PD (11,0)	470
JBPUTA	The total amount of user and control data that was sent by the user's program. This value does not include the LLID, MAPNAME, or FMH-7 data lengths.	PD (11,0)	476
JBGETN	The number of times ACGET was called to receive user or control data. Calls that result in no data being given to the user application will not be counted.	PD (11,0)	482
JBGETA	The total amount of user and control data that was received by the user's program. This value does not include the LLID, MAPNAME, or FMH-7 data lengths.	PD (11,0)	488
JBPGIN	The number of intervals that begin at the first put of a chain and end when CD is returned to the user.	PD (11,0)	494
JBPGIL	The amount of time (in milliseconds) spent in intervals that begin at the first put of a chain and end when CD is returned to the user.	PD (11,0)	500

Figure A-4 (Page 5 of 5). Job Data (One Entry for Each Job in the System, Collected for Each Interval)

Field Name	Description	Attributes	Buffer Position
JBGGIL	The amount of time (in milliseconds) spent in intervals that begin when the first get of a get chain completes and ends when the first get of a new chain is issued.	PD (11,0)	506
JBRTI	This is the number of request I/O commands (REQIOs) issued to transmit data of any kind (including FMH-7s).	PD (11,0)	512
JBRRI	This is the number of REQIOs issued to receive data of any kind (including FMH-7s).	PD (11,0)	518

Notes:

1. Job Types:
   - B = Batch
   - I = Interactive
   - A = Autostart
   - R = Spool reader
   - W = Spool writer
   - M = Subsystem monitor
   - S = System
   - X = SCPF job
   - H = HMC microcode
   - V = VMC microcode
2. Job Subtypes:
   - T = MRT (System/36 environment only)
   - E = Evoke (communications batch)
   - P = Print driver job
   - J = Prestart job
3. Job Flags:

Bit	
0	Pass-through source
1	Pass-through target
2	Emulation active
3	PC Support application
4	Target DDM job
5	MRT
6-15	Not used

File Name: QAPMDISK

Disk File Entries: In Figure A-5, reference is made to a disk arm, or actuator, as opposed to a disk drive. The 9332 offers two configurations: 200MB and 400MB. The 200MB version has one enclosure (one disk drive) with one actuator. The 400MB version has one enclosure (one disk drive) with two actuators. The 9335 has one size: 850MB with 2 actuators. The 9337 has three size configurations, each with one actuator per disk. The available sizes are 542MB, 970MB, and 1967MB.

Figure A-5 lists the fields in the disk data file.

Figure A-5 (Page 1 of 5). Disk Storage Data (One for Each Actuator)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4

Figure A-5 (Page 2 of 5). Disk Storage Data (One for Each Actuator)

Field Name	Description	Attributes	Buffer Position
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
DIOPID	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
DSARM	Disk arm number: Specifies the unique identifier of the unit. Each actuator arm on the disk drives available to the machine represents a unit of auxiliary storage. The value of the unit number is assigned by the system when the unit is allocated to an ASP.	C (4)	21
DSTYPE	Disk drive type such as 9332, 9335, or 6100.	C (4)	25
DSSCAN	Number of search string commands: This count is always zero, because search string commands are not supported for the 9332, 9335, or 6100.	PD (5,0)	29
DSBLKR	Number of blocks read: The block length is 520 bytes, which includes 8 bytes of system control information.	PD (11,0)	32
DSBLKW	Number of blocks written: The block length is 520 bytes, which includes 8 bytes of system control information.	PD (11,0)	38
DSIDLC	Processor idle loop counter (see Note 1 on page A-34): The number of times the disk controller passed through the idle loop. This count is increased differently for the 9332 and the 9335. For the 9332, this counter is increased only if the disk controller is totally idle (for example, no I/O operations are active). For the 9335, even though the disk controller may be idle and the counter gets increased, an I/O operation can be active (for example, seek is being performed). This field is 0 for disk drive type 6100.	PD (11,0)	44
DSIDLT	Processor idle loop time (see Note 4 on page A-34): The time (in hundredths of microseconds) to make one pass through the idle loop. This field is 0 for disk drive type 6100.	PD (11,0)	50
DSSK1	Number of seeks > 2/3 (see Note 2 on page A-34): The number of times the arm traveled more than 2/3 of the disk on a seek.	PD (11,0)	56
DSSK2	Number of seeks > 1/3 and < 2/3 (see Note 2 on page A-34): The number of times the arm traveled more than 1/3 but less than 2/3 of the disk on a seek.	PD (11,0)	62
DSSK3	Number of seeks > 1/6 and < 1/3 (see Note 2 on page A-34): The number of times the arm traveled more than 1/6 but less than 1/3 of the disk on a seek.	PD (11,0)	68
DSSK4	Number of seeks > 1/12 and < 1/6 (see Note 2 on page A-34): The number of times the arm traveled more than 1/12 but less than 1/6 of the disk on a seek.	PD (11,0)	74
DSSK5	Number of seeks < 1/12 (see Note 2 on page A-34): The number of times the arm traveled from its current position but less than 1/12 of the disk on a seek.	PD (11,0)	80

Figure A-5 (Page 3 of 5). Disk Storage Data (One for Each Actuator)

Field Name	Description	Attributes	Buffer Position
DSSK6	Number of zero seeks (see Note 1 on page A-34): The number of times the access arm did not physically move on a seek request. The operation may have resulted in a head switch. This field is 0 for disk drive type 6100. The number of zero seeks will be accumulated in DSSK5.	PD (11,0)	86
DSQUEL	Total queue elements (see Note 5 on page A-34): The number of I/O operations waiting service at sample time. This number includes the I/O operation that is in progress. Divide this by DSSMPL to get the average queue length.	PD (11,0)	92
DSNBSY	Number of times arm not busy (see Note 5 on page A-34): The number of times there were no outstanding I/O operations active at sample time.	PD (11,0)	98
DSSMPL	Number of samples taken at two per second (see Note 5 on page A-34): The number of samples taken at approximately two per second for the DSQUEL and DSNBSY fields.	PD (11,0)	104
DSCAP	Drive capacity (in bytes): Total number of bytes of auxiliary storage provided on the unit for the storage of objects and internal machine functions when the ASP containing it is not under checksum protection. The unit reserved system space value is subtracted from the unit capacity to calculate this capacity. When the ASP containing the unit is under checksum protection, the unit checksum information describes how much of the unit capacity is used for protected space, unprotected space, and redundant data.	PD (11,0)	110
DSAVL	Drive available space (in bytes): Total number of bytes of auxiliary storage space that is not currently assigned to objects or internal machine functions, and therefore is available on the unit if the ASP containing it is not under checksum protection.	PD (11,0)	116
DSASP	ASP number: Specifies the ASP to which this unit is currently allocated. A value of 1 specifies the system ASP. A value from 2 through 16 specifies a user ASP. A value of 0 indicates that this unit is currently not allocated.	C (2)	122
DSCSS	Checksum number: Specifies the checksum set to which this unit is currently allocated. A value from 1 through 16 specifies a checksum set. A value of 0 specifies that the unit is currently not assigned to a checksum set.	C (2)	124
DSPCAP	Permanent storage capacity: Number of bytes of auxiliary storage formatted for storage of protected data on the unit. This field has a value other than zero if this unit is allocated to a checksum set. Units not allocated to a checksum set contain no permanent storage area. This value does not include the size of any data redundancy area which may also be formatted on the unit.	PD (11,0)	126
DSPAVL	Permanent storage space available: Number of bytes of permanent space on auxiliary storage available for allocation on the unit that is not currently assigned to objects of internal machine functions. This field has a value other than zero only if this unit is allocated to a checksum set. Units not allocated to a checksum set contain no protected storage area.	PD (11,0)	132
DSIOPB	IOP bus number.	PD (3,0)	138
DSIOPA	IOP bus address.	PD (3,0)	140

Figure A-5 (Page 4 of 5). Disk Storage Data (One for Each Actuator)

Field Name	Description	Attributes	Buffer Position
DSPORT	Device port number.	PD (3,0)	142
DSARMP	Arm number. For the 9332, this can be 0 or 1. For the 9335, this can be 0 to 7. For the 6100, this can be 0.	PD (3,0)	144
DMFLAG	' ' means this arm is not mirrored. 'A' means this is the designated first arm of a mirrored pair. 'B' means this is the designated second arm of a mirrored pair.	C (1)	146
DMSTS	File mirroring status. 1 = active, 2 = resuming, 3 = suspended	PD (1,0)	147
DMIOPB	IOP bus number the arm is on.	PD (3,0)	148
DMIOPA	IOP bus address of the arm's mirror.	PD (3,0)	150
DMPORT	Port number of the arm's mirror.	PD (3,0)	152
DMARMP	Arm number of the arm's mirror.	PD (3,0)	154
DSRDS	Number of read data commands.	PD (11,0)	156
DSWRTS	Number of write data commands.	PD (11,0)	162
DSBUFO	Number of buffer overruns: The number of times that data was available to be read into the disk controller buffer from the disk, but the disk controller buffer still contained valid data that was not retrieved by the storage device controller. Consequently, the disk had to take an additional revolution until the buffer was available to accept data. This field is 0 for disk drive type 6100.	PD (11,0)	168
DSBUFU	Number of buffer underruns: The number of times that the disk controller was ready to transfer data to the disk on a write, but the disk controller buffer was empty. The data was not transferred in time by the disk IOP to the disk controller buffer. The disk was forced to take an extra revolution awaiting the data. This field is 0 for disk drive type 6100.	PD (11,0)	174
DSMDLN	Model Number: The model number of the disk drive.	C (4)	180
DSDCRH	Device cache read hits: The number of times that all of the data requested by the read operation was obtained from the device read or write cache.	PD (11,0)	184
DSDCPH	Device cache partial read hits: The number of times that a portion, but not all, of the data requested by the read operation was obtained by the device read or write cache. A physical operation to the device media was required to obtain the remaining data.	PD (11,0)	190
DSDCWH	Device cache write hits: The number of times that the data associated with a write operation replaces, or is combined with, existing data in the device write cache, thereby eliminating a write operation.	PD (11,0)	196
DSDCFW	Device cache fast writes: The number of times that space was available in the device write cache for the data associated with a write operation and a response was returned immediately.	PD (11,0)	202
DSDROP	Device read operations: The number of read operations issued to the device by the controller. This includes operations generated for redundant system data areas. It does not include operations generated for diagnostics or access to the controller reserved area that occur during this idle time.	PD (11,0)	208

Figure A-5 (Page 5 of 5). Disk Storage Data (One for Each Actuator)

Field Name	Description	Attributes	Buffer Position
DSDWOP	Device write operations: The number of write operations issued to the device by the controller. This includes operations generated for redundant system data areas. It does not include operations generated for diagnostics or access to the controller reserved area that occur during this idle time.	PD (11,0)	214
	Reserved.	PD (11,0)	220
	Reserved.	PD (11,0)	226
	Reserved.	PD (11,0)	232
	Reserved.	PD (11,0)	238

Notes:

1. 9332/9335 inconsistencies:
   - 9335 updates the idle count only when the processing unit (A) is not busy. Disk operations such as seek could be in progress. 9332 updates the idle count when there is no activity in any of its processors.
   - If there is no movement and no head switch, the 9332 does not count this operation as a zero seek, the 9335 does.
   - If an operation causes a head switch (starts a read or write on one track and end up on another track), the 9332 counts this as a zero seek, the 9335 does not.
2.
```
9335:  > 2/3                    9332:  >= 2/3
       > 1/3  and <= 2/3                >= 1/3  and < 2/3
       > 1/6  and <= 1/3                >= 1/6  and < 1/3
       > 1/12 and <= 1/6                >= 1/12 and < 1/6
                  <= 1/12                               <1/12
```
3. DSIDLC and DSIDLT are duplicated across all arms for the same disk controller.
4. The idle loop count and time are used to calculate the storage device controller utilization as follows:

   Convert the product of the idle loop count times the idle loop time from hundredths of microseconds to seconds. Subtract this from the interval time, and divide the result by the interval time. For example:

   ```
   IOP utilization = (INTSEC - (DSIDLC x DSIDLT)/10**8)/ INTSEC
   ```
5. The following three counts are extracted from the QAPMDISK record for AS/400 9406 system only:
   - DSSMPL
   - DSQUEL
   - DSNBSY
6. The following formulas describe how several of the fields in the previous table can be used to calculate utilization and service time for each arm.

   Arm utilization (DSUTL): That part of the total interval that the arm was being used for I/O operations.

   ```
   DSUTL = Arm Busy = (DSSMPL - DSNBSY)/DSSMPL
   ```

   Arm accesses per second (DSAS): The number of reads and writes per second for this arm during the interval.

   ```
   DSAS = (DSRDS + DSWRTS)/INTSEC
   ```

   Service time (DSSRVCT): The average time for an arm I/O operation. This includes disk controller time.

   ```
   DSSRVCT = DSUTL/DSAS
   ```

File Name: QAPMPOOL

Main Storage Pool File Entries:

Figure A-6 lists the fields in the storage pool file.

Figure A-6. Main Storage Data (One for Each Storage Pool)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
PONBR	Pool number: Specifies the unique identifier of this pool. The value is from 1 to 16.	C (2)	20
POACTL	Pool activity level setting: The maximum number of processes that can be active in the machine at the same time.	PD (3,0)	22
POSIZ	Pool size (in KB): The amount of main storage assigned to the pool.	PD (7,0)	24
PORES	Pool reserved size (in KB): Specifies the amount of storage from the pool that is dedicated to machine functions.	PD (5,0)	28
PODBF	Pool database faults: Total number of interruptions to processes (not necessarily assigned to this pool) that were required to transfer data into the pool to permit the MI instruction to process the database function.	PD (11,0)	31
PONDBF	Pool nondatabase faults: Total number of interruptions to processes (not necessarily assigned to this pool) that were required to transfer data into the pool to permit the MI instruction to process nondatabase functions.	PD (11,0)	37
PODBPG	Pool database pages read: Total number of pages of database data transferred from auxiliary storage to the pool to permit the instruction to run as a consequence of set access state, implicit access group movement, and internal machine actions.	PD (11,0)	43
PONDPG	Pool nondatabase pages read: Total number of pages of database data transferred from auxiliary storage to the pool to permit the instruction to run as a consequence of set access state, implicit access group movement, and internal machine actions.	PD (11,0)	49
POAW	Number of active to wait transitions: Total number of transitions by processes assigned to this pool from active state to wait state.	PD (11,0)	55
POWI	Number of wait to ineligible: Total number of transitions by processes assigned to this pool from wait state to ineligible state.	PD (11,0)	61
POAI	Number of active to ineligible: Total number of transitions by processes assigned to this pool from active state to ineligible state.	PD (11,0)	67

File Name: QAPMHDLC

HDLC File Entries: In Figure A-7, statistics are kept on a line basis for the fields in the high-level data link control (HDLC) file.

Figure A-7 (Page 1 of 3). SDLC/HDLC Statistics

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: the nth sample interval since the start of the performance monitor job.	PD (5,0)	1

Figure A-7 (Page 2 of 3). SDLC/HDLC Statistics

Field Name	Description	Attributes	Buffer Position
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
SHIOP	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
SHLND	Line description: The name of the description for this line.	C (10)	21
	Reserved.	C (10)	31
SHLSP	Line speed: The speed of the line in bits per second (bps).	PD (11,0)	41
SHBTRN	Bytes transmitted: The number of bytes transmitted including bytes transmitted again.	PD (11,0)	47
SHBRCV	Bytes received: The number of bytes received including all bytes in frames that had any kind of error.	PD (11,0)	53
SHPRCL	Protocol type: S for SDLC.	C (1)	59
SHFTRN	Number of frames transmitted (I, supervisory, and frames not numbered) excluding frames transmitted again.	PD (11,0)	60
SHIFTR	Number of I-frames transmitted excluding I-frames transmitted again.	PD (11,0)	66
SHIFRT	Number of I-frames transmitted again.	PD (11,0)	72
SHFRT	Number of I, supervisory, and frames not numbered transmitted again.	PD (11,0)	78
SHEFFR	Error-free frames received: The number of I, supervisory, and frames not numbered received without error (whether or not they were transmitted again from the remote side).	PD (11,0)	84
SHEFIR	Error-free I-frames received: The number of I-frames received without error (whether or not they were transmitted again from the remote side).	PD (11,0)	90
SHFRIE	Frames received in error: The number of I, supervisory, and frames not numbered received in error. There are three error possibilities: (1) a supervisory or I-frame was received with an Nr count that is requesting retransmission of a frame, (2) an I-frame was received with an Ns count that indicates that frames were missed, (3) a frame is received with one of the following errors–a frame check sequence error, an abnormal end, a receive overrun, or a frame truncated error.	PD (11,0)	96
SHIFR	Invalid frames received: The number of not valid frames received. These are frames received with either: (1) short frame error–frame is less than 32 bits or (2) residue error–frame is not on a byte boundary.	PD (11,0)	102
SHRRFT	Number of receive ready supervisory frames transmitted.	PD (11,0)	108
SHRRFR	Number of receive ready supervisory frames received.	PD (11,0)	114
SHRNRT	Number of receive not ready supervisory frames transmitted.	PD (11,0)	120
SHRNRR	Number of receive not ready supervisory frames received.	PD (11,0)	126

Figure A-7 (Page 3 of 3). SDLC/HDLC Statistics

Field Name	Description	Attributes	Buffer Position
SHLNKR	Data link resets: The number of times a set normal response mode (SNRM) was received when the station was already in normal response mode.	PD (11,0)	132
SHIOPB	IOP bus number.	PD (3,0)	138
SHIOPA	IOP bus address.	PD (3,0)	140
SHCPT	The length of time (in tenths of seconds) that the system waits for the response to a poll while in normal disconnect mode before polling the next station.	PD (3,0)	142

File Name: QAPMASYN

Asynchronous File Entries: Figure A-8 lists the fields in the asynchronous file.

Figure A-8. Asynchronous Statistics (One Per Line)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
AIOPID	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
ASLND	Line description: The name of the description for this line.	C (10)	21
	Reserved.	C (10)	31
ASLSP	Line speed: The speed of this line in bits per second (bps).	PD (11,0)	41
ASBTRN	Number of bytes transmitted (data and control characters) including bytes transmitted again because of errors.	PD (11,0)	47
ASBRCV	Number of bytes received (data and control characters), including characters received in error.	PD (11,0)	53
ASPRCL	Protocol type: A for asynchronous.	C (1)	59
ASPDUR	The total number of protocol data units received.	PD (11,0)	60
ASPDUE	The total number of protocol data units received with parity and stop bit errors.	PD (11,0)	66
ASPDUT	The total number of protocol data units successfully transmitted and the data-circuit ending equipment (DCE) acknowledged.	PD (11,0)	72
ASIOPB	IOP bus number.	PD (3,0)	78
ASIOPA	IOP bus address.	PD (3,0)	80

File Name: QAPMBSC

Binary Synchronous File Entries: Figure A-9 lists the fields in the binary synchronous file.

Figure A-9 (Page 1 of 2). BSC Statistics (One Per Link/Station)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
BIOPID	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
BSLND	Line description: The name of the description for this line.	C (10)	21
	Reserved.	C (10)	31
BSLSP	Line speed: The speed of the line in bits per second (bps).	PD (11,0)	41
BSBTRN	Bytes transmitted: The number of bytes (data and control characters) transmitted, including bytes transmitted again.	PD (11,0)	47
BSBRCV	Bytes received: The number of bytes (data and control characters) received including bytes received in error.	PD (11,0)	53
BSPRCL	Protocol type: B for binary synchronous.	C (1)	59
BSDCRV	Data characters received: The number of data characters received successfully (excluding synchronous characters) while in data mode. For feature types 2507 and 6150, this value is equal to field BSBRCV.	PD (11,0)	60
BSDCRE	Data characters received in error: The number of data characters received with a block-check character error while in data mode. For feature types 2507 and 6150, this value is equal to field BSCRER.	PD (11,0)	66
BSDCTR	Data characters transmitted: The number of data characters transmitted successfully while in data mode. For feature types 2507 and 6150, this value is equal to field BSBTRN.	PD (11,0)	72
BSCRER	Characters received in error: The number of characters received with a block-check character error.	PD (11,0)	78
BSLNK	Negative acknowledgment character received to text sent (see Note). The number of times the remote station or device did not understand the command sent from the host system.	PD (11,0)	84
BSLWA	Wrong acknowledgment character to text sent (see Note). The host system received an acknowledgment from the remote device that was not expected. For example, the system expected an ACK0 and received an ACK1.	PD (11,0)	90
BSLQTS	Enqueue to text sent (see Note): Text was sent by a station and an ENQ character was returned. The receiving station expected some form of acknowledgment, such as an ACK0, ACK1, or NAK.	PD (11,0)	96

Figure A-9 (Page 2 of 2). BSC Statistics (One Per Link/Station)

Field Name	Description	Attributes	Buffer Position
BSLINV	Invalid (unrecognized format): One of the delimiter characters that encloses the data in brackets being sent/received is not valid (see Note).	PD (11,0)	102
BSLQAK	Enqueue to acknowledged character: The remote station returned an acknowledgment (for example, ACK0) and the host system sent an ENQ character. This indicates that the host station did not recognize the acknowledgment as a valid acknowledgment (see Note).	PD (11,0)	108
BSLTNK	Negative acknowledgment character received to text sent (total): The number of times the remote station did not understand the command sent from the host system (see Note).	PD (11,0)	114
BSLTWA	Wrong acknowledgment character to text sent (total): The host system received an acknowledgment from the remote device that was not expected. For example, the host system expected an ACK0 and received an ACK1 (see Note).	PD (11,0)	120
BSLTQT	Enqueue to text sent (total): Text was sent by a station and an ENQ character was returned. The receiving station expected some form of acknowledgment such as an ACK0, ACK1, or NAK (see Note).	PD (11,0)	126
BSLTIV	Invalid (unrecognized format) (total): One of the delimiter characters that enclose the data in brackets being sent/received is not valid (see Note).	PD (11,0)	132
BSLTQA	Enqueue to acknowledged character (total): The remote station returned an acknowledgment (for example, ACK0) and the host station sent an ENQ character. This indicates that the host station did not recognize the acknowledgment as a valid acknowledgment (see Note).	PD (11,0)	138
BSLDRA	Disconnect received: The remote station issued a disconnect with abnormal end. This could occur when error recovery did not succeed or the binary synchronous job was ended.	PD (11,0)	144
BSLEAB	End of transmission (EOT) received (abnormal end): Similar to a disconnect.	PD (11,0)	150
BSLDFA	Disconnect received (forward abnormal end): The host station issued a disconnect with abnormal end. This could occur when the error recovery did not succeed, or the binary synchronous job was ended.	PD (11,0)	156
BSLEFA	EOT received (forward abnormal end): Similar to a disconnect.	PD (11,0)	162
BSLDBT	Number of data blocks transmitted.	PD (11,0)	168
BSLDBR	Number of data blocks received.	PD (11,0)	174
BSLBKR	Number of data blocks transmitted again.	PD (11,0)	180
BSLBKE	Number of data blocks received in error.	PD (11,0)	186
BSLTRT	Total number of characters transmitted again, including control characters.	PD (11,0)	192
BSLDRT	Total number of data characters transmitted again.	PD (11,0)	198
BSIOPB	IOP bus number.	PD (3,0)	204
BSIOPA	IOP bus address.	PD (3,0)	206

Note: The counters BSLNK through BSLQAK are error recovery counters, and are increased the first time the error is detected. The counters BSLTNK to BSLTQA are error recovery counters, and are increased every time the error occurs. The same errors are being counted in each set of counters, so the first set indicates how many times an error was detected, and the second set indicates how many retries it took to recover from the errors.

File Name: QAPMX25

X.25 File Entries: Figure A-10 lists the fields in the X.25 file.

In Figure A-10, the XH prefix in the label refers to HDLC counters, XL refers to X.25 logical link control (LLC) counters, and XP refers to packet level control (PLC) counters.

Figure A-10 (Page 1 of 3). X.25 Statistics

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
XIOPID	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
XLLND	Line description: The name of the description for this line.	C (10)	21
	Reserved.	C (10)	31
XLLSP	Line speed: The speed of this line in bits per second (bps).	PD (11,0)	41
XHBTRN	Bytes transmitted: The number of bytes transmitted, including bytes transmitted again.	PD (11,0)	47
XHBRCV	Bytes received: The number of bytes received, including all bytes in frames that had any kind of error.	PD (11,0)	53
XHPRCL	Protocol type: X for X.25.	C (1)	59
XHFTRN	Frames transmitted: The number of frames transmitted (I, supervisory, and frames not numbered), excluding frames transmitted again.	PD (11,0)	60
XHIFTR	I-frames transmitted: The number of I-frames transmitted, excluding I-frames transmitted again.	PD (11,0)	66
XHIFRT	I-frames transmitted again: The number of I-frames transmitted again.	PD (11,0)	72
XHFRT	Frames transmitted again: The number of I, supervisory, and frames not numbered transmitted again.	PD (11,0)	78
XHEFFR	Error-free frames received: The number of I, supervisory, and frames not numbered received without error (whether or not they were transmitted again from the remote side).	PD (11,0)	84
XHEFIR	Error-free I-frames received: The number of I-frames received without error (whether or not they were transmitted again from the remote side).	PD (11,0)	90

Figure A-10 (Page 2 of 3). X.25 Statistics

Field Name	Description	Attributes	Buffer Position
XHFRIE	Frames received in error: The number of I, supervisory, and frames not numbered received in error. There are three error possibilities: (1) a supervisory or I-frame was received with an Nr count that is requesting retransmission of a frame, (2) an I-frame was received with an Ns count that indicates that frames were missed, (3) a frame was received with one of the following errors–a frame check sequence error, an abnormal end, a receive overrun or a frame truncated error.	PD (11,0)	96
XHIFR	Invalid frames received: The number of not valid frames received. These are frames received with either: (1) a short frame error–frame is less than 32 bits, or (2) a residue error–frame is not on a byte boundary.	PD (11,0)	102
XHRRFT	Number of receive ready supervisory frames transmitted.	PD (11,0)	108
XHRRFR	Number of receive ready supervisory frames received.	PD (11,0)	114
XHRNRT	Number of receive-not-ready supervisory frames transmitted.	PD (11,0)	120
XHRNRR	Number of receive-not-ready supervisory frames received.	PD (11,0)	126
XHLNKR	Link resets: The number of times when a set normal response mode (SNRM) was received when the station was already in normal response mode.	PD (11,0)	132
	Reserved.	C (2)	138
	Reserved.	C (2)	140
	Reserved.	C (2)	142
XLITR	Interface protocol data units transmitted (LLC level).	PD (11,0)	144
XLIRC	Interface protocol data units received.	PD (11,0)	150
XLIRT	Interface protocol data units transmitted again.	PD (11,0)	156
XLIRE	Interface protocol data units received in error (checksum).	PD (11,0)	162
XLLXTR	Number of XIDS transmitted.	PD (11,0)	168
XLXRC	Number of XIDS received.	PD (11,0)	174
XLTT	Number of tests transmitted.	PD (11,0)	180
XLTR	Number of tests received.	PD (11,0)	186
XLLJT	Number of LLC rejects transmitted.	PD (11,0)	192
XLLJR	Number of LLC rejects received.	PD (11,0)	198
XLRLD	Number of received LLC protocol data units discarded.	PD (11,0)	204
XLTO	Number of time-outs.	PD (11,0)	210
XLCED	Checksum errors detected.	PD (11,0)	216
XLSRA	Successful recovery attempts.	PD (11,0)	222
XLRA	Recovery attempts.	PD (11,0)	228
XLRSI	Number of reset indications from packet-link control.	PD (11,0)	234
XLCLS	Number of close station indications from packet-link control.	PD (11,0)	240
XLRNR	LLC receive-not-ready frames received.	PD (11,0)	246
	Reserved.	C (2)	252
	Reserved.	C (1)	254

Figure A-10 (Page 3 of 3). X.25 Statistics

Field Name	Description	Attributes	Buffer Position
XPTPT	Total packets transmitted.	PD (11,0)	255
XPTPR	Total packets received.	PD (11,0)	261
XPDPT	Data packets transmitted.	PD (11,0)	267
XPDPR	Data packets received.	PD (11,0)	273
XPRPT	Reset packets transmitted.	PD (11,0)	279
XPROR	Reset packets received.	PD (11,0)	285
XPRRT	Reserved.	PD (11,0)	291
XPRIR	Reserved.	PD (11,0)	297
XPRNR	Receive-not-ready packets received.	PD (11,0)	303
XIOPB	IOP bus number.	PD (3,0)	309
XIOPA	IOP bus address.	PD (3,0)	311

File Name: QAPMECL

Token-Ring Network File Entries: Figure A-11 lists the fields in the token-ring local area network file.

Figure A-11 (Page 1 of 3). Token-Ring Network Counter

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
EIOPI	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
ELLND	Line description: The name of the description for this line.	C (10)	21
	Reserved.	C (10)	31
ELLSP	Line speed: The line speed expressed in bits per second (bps).	PD (11,0)	41
ELTFT	Total number of Type II frames transmitted.	PD (11,0)	47
ELTFR	Total number of Type II frames received.	PD (11,0)	53
ELIFT	Total number of I-frames transmitted.	PD (11,0)	59
ELIFR	Total number of I-frames received.	PD (11,0)	65
ELICT	Total number of characters transmitted in all I-frames.	PD (11,0)	71
ELICR	Total number of characters received in all I-frames.	PD (11,0)	77

Figure A-11 (Page 2 of 3). Token-Ring Network Counter

Field Name	Description	Attributes	Buffer Position
ELPRCL	Protocol type: E for token-ring network.	C (1)	83
ELRFT	Number of receive-not-ready frames transmitted.	PD (5,0)	84
ELRFR	Number of receive-not-ready frames received.	PD (5,0)	87
ELFFT	Number of frame-reject frames transmitted.	PD (5,0)	90
ELFFR	Number of frame-reject frames received.	PD (5,0)	93
ELRJFR	Number of reject frames received.	PD (5,0)	96
ELRJFT	Number of reject frames transmitted.	PD (5,0)	99
ELSFT	Number of set asynchronous balanced mode extended frames transmitted.	PD (5,0)	102
ELSFR	Number of set asynchronous balanced mode extended frames received.	PD (5,0)	105
ELDFT	Number of disconnect frames transmitted.	PD (5,0)	108
ELDFR	Number of disconnect frames received.	PD (5,0)	111
ELDMT	Number of disconnect mode frames transmitted.	PD (5,0)	114
ELDMR	Number of disconnect mode frames received.	PD (5,0)	117
ELN2R	N2 retries end count: This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	120
ELT1T	T1 timer end count: Number of times the T1 timer ended. This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	123
EMFTR	Total frames transmitted: Total number of frames (LLC and MAC) transmitted.	PD (11)	126
EMFRV	Total frames received: Total number of frames (LLC and MAC) received.	PD (11)	132
EMMFT	MAC frames transmitted: Total number of MAC frames transmitted.	PD (11)	138
EMMFR	MAC frames received: Total number of MAC frames received.	PD (11)	144
EMRIT	Routing information frames transmitted: Total number of frames (LLC and MAC) with a routing-information field transmitted.	PD (11)	150
EMRIR	Routing information frames received: Total number of frames (LLC and MAC) with a routing-information field received.	PD (11)	156
EMLNE	Line error: Code violation of frame-check sequence error.	PD (5,0)	162
EMINE	Internal error: Adapter internal error.	PD (5,0)	165
EMBRE	Burst error: Burst of same polarity is detected by the physical unit after the starting delimiter of a frame or token.	PD (5,0)	168
EMAFE	Address-recognized indicator or frame-copied indicator error: Physical control field-extension field error.	PD (5,0)	171
EMABT	Abnormal ending delimiter: Abnormal ending delimiter transmitted because of internal error.	PD (5,0)	174
EMLST	Lost frame: Physical trailer timer ended while IOA is in transmit stripping state.	PD (5,0)	177

Figure A-11 (Page 3 of 3). Token-Ring Network Counter

Field Name	Description	Attributes	Buffer Position
EMRXC	Receive congestion: Frame not copied because no buffer was available for the IOA to receive.	PD (5,0)	180
EMFCE	Frame-copied error: The frame with a specific destination address was copied by another adapter.	PD (5,0)	183
EMFQE	Frequency error on the adapter.	PD (5,0)	186
EMTKE	Token error: The adapter any token timer ended without detecting any frame or token.	PD (5,0)	189
EMDBE	Direct memory access bus error: IOP/IOA bus DMA error.	PD (5,0)	192
EMDPE	Direct memory access parity error: IOP/IOA DMA parity error.	PD (5,0)	195
EMANR	Total number of frames with address not recognized error.	PD (5,0)	198
EMFNC	Total number of frames with frame not copied error.	PD (5,0)	201
EMTSE	Total number of adapter frame transmit or frame strip process errors.	PD (5,0)	204
EMUAP	Unauthorized access priority: The access priority requested is not authorized.	PD (5,0)	207
EMUMF	Unauthorized MAC frame: The adapter is not authorized to send a MAC frame with the source class specified, or the MAC frame has a source class of zero, or the MAC frame physical control field attention field is > 1.	PD (5,0)	210
EMSFT	Soft error: Total number of soft errors as reported by the adapter.	PD (5,0)	213
EMTBC	Total number of beacon frames transmitted.	PD (5,0)	216
EMIOA	IOA status overrun: Adapter interrupt status queue overrun, earliest status discarded.	PD (5,0)	219
EMFDC	Total number of frames discarded.	PD (11)	222
EMSIN	Total number of interrupts that MAC could not decode.	PD (11,0)	228
EIOPB	IOP bus number.	PD (3,0)	234
EIOPA	IOP bus address.	PD (3,0)	236

File Name: QAPMSTNL

Token-Ring Station File Entries: Figure A-11 lists the fields in the token-ring local area network station file.

Figure A-12 (Page 1 of 3). Token-Ring Station Counter

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16

Figure A-12 (Page 2 of 3). Token-Ring Station Counter

Field Name	Description	Attributes	Buffer Position
SLIOPI	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
SLPCEP	The provider connection end point (PCEP) ID.	C (8)	21
SLLND	Line description: The name of the description for this line.	C (10)	29
SLSTNN	Station name: The name of the station on this line.	C (10)	39
SLLSPD	Line speed: The line speed expressed in bits per second (bps).	PD (11,0)	49
SLTXMT	Total number of Type II frames transmitted.	PD (11,0)	55
SLTRCV	Total number of Type II frames received.	PD (11,0)	61
SLBXMT	Total number of bytes transmitted in all I-frames.	PD (11,0)	67
SLBRCV	Total number of bytes received in all I-frames.	PD (11,0)	73
SLIXMT	Total number of I-frames transmitted.	PD (11,0)	79
SLIRCV	Total number of I-frames received.	PD (11,0)	85
SLIREX	Number of I-frames retransmitted.	PD (11,0)	91
SLBREX	Number of bytes retransmitted in I-frames.	PD (11,0)	97
SLRNRX	Number of receive-not-ready frames transmitted.	PD (5,0)	103
SLRNRR	Number of receive-not-ready frames received.	PD (5,0)	106
SLFRMX	Number of frame-reject frames transmitted.	PD (5,0)	109
SLFRMR	Number of frame-reject frames received.	PD (5,0)	112
SLREJR	Number of reject frames received.	PD (5,0)	115
SLREJT	Number of reject frames transmitted.	PD (5,0)	118
SLSABX	Number of set asynchronous balanced mode extended frames transmitted.	PD (5,0)	121
SLSABR	Number of set asynchronous balanced mode extended frames received.	PD (5,0)	124
SLDISX	Number of disconnect frames transmitted.	PD (5,0)	127
SLDISR	Number of disconnect frames received.	PD (5,0)	130
SLDMFX	Number of disconnect mode frames transmitted.	PD (5,0)	133
SLDMFR	Number of disconnect mode frames received.	PD (5,0)	136
SLN2RE	N2 retries end count: This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	139
SLT1TE	T1 timer end count: Number of times the T1 timer ended. This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	142
SLTITE	Ti timer end count: Number of times the Ti timer (inactivity timer) ended.	PD (5,0)	145

Figure A-12 (Page 3 of 3). Token-Ring Station Counter

Field Name	Description	Attributes	Buffer Position
SLLBCT	Local busy count: Number of times station entered local busy sub-state.	PD (5,0)	148
SLPRCL	Protocol type: E for token-ring network.	C (1)	151
SLIOPB	IOP bus number.	PD (3,0)	152
SLIOPA	IOP bus address.	PD (3,0)	154

File Name: QAPMETH

Ethernet File Entries: Figure A-13 lists the fields in the Ethernet file.

Figure A-13 (Page 1 of 3). Ethernet Counter

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
ETIOPI	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
ETLLND	Line description: The name of the description for this line.	C (10)	21
	Reserved.	C (10)	31
ETLLSP	Line speed: The line speed expressed in bits per second (bps).	PD (11,0)	41
ETLTFT	Total number of Type II frames transmitted.	PD (11,0)	47
ETLTFR	Total number of Type II frames received.	PD (11,0)	53
ETLIFT	Total number of I-frames transmitted.	PD (11,0)	59
ETLIFR	Total number of I-frames received.	PD (11,0)	65
ETLICT	Total number of characters transmitted in all I-frames.	PD (11,0)	71
ETLICR	Total number of characters received in all I-frames.	PD (11,0)	77
ETLPRCL	Protocol type: T for Ethernet.	C (1)	83
ETLRFT	Number of receive-not-ready frames transmitted.	PD (5,0)	84
ETLRFR	Number of receive-not-ready frames received.	PD (5,0)	87
ETLFFT	Number of frame-reject frames transmitted.	PD (5,0)	90
ETLFFR	Number of frame-reject frames received.	PD (5,0)	93
ETLRJFR	Number of reject frames received.	PD (5,0)	96
ETLRJFT	Number of reject frames transmitted.	PD (5,0)	99

Figure A-13 (Page 2 of 3). Ethernet Counter

Field Name	Description	Attributes	Buffer Position
ETLSFT	Number of set asynchronous balanced mode extended frames transmitted.	PD (5,0)	102
ETLSFR	Number of set asynchronous balanced mode extended frames received.	PD (5,0)	105
ETLDFT	Number of disconnect frames transmitted.	PD (5,0)	108
ETLDFR	Number of disconnect frames received.	PD (5,0)	111
ETLDMT	Number of disconnect mode frames transmitted.	PD (5,0)	114
ETLDMR	Number of disconnect mode frames received.	PD (5,0)	117
ETLN2R	N2 retries end count: This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	120
ETLT1T	T1 timer end count: Number of times the T1 timer ended. This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	123
ETLTIT	Number of times the TI timer (Inactivity Timer) expired. This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	126
ETLFTR	Number of times I-frame retransmission occurred.	PD (5)	129
ETLBRT	Total number of bytes transmitted.	PD (5)	132
ETLLBC	Local busy count: Number of times station entered local busy substate.	PD (5)	135
ETMFTG	Frames transmitted without error.	PD (11)	138
ETMFRG	Frames received without error.	PD (11)	144
ETMIFM	Inbound frames missed: A receiver buffer error or a missed frame was detected by the IOA.	PD (5)	150
ETMCRE	CRC error: Checksum errors detected by the receiver.	PD (5,0)	153
ETMEXR	More than 16 retries: Frame unsuccessfully transmitted due to excessive retries.	PD (5,0)	156
ETMOWC	Out of window collisions: Collision occurred after slot time of channel elapsed.	PD (5,0)	159
ETMALE	Alignment error: Inbound frame contained non-integer number of bytes and a CRC error.	PD (5,0)	162
ETMCRL	Carrier loss: Carrier input to the chipset on the IO adapters is false during transmission.	PD (5,0)	165
ETMTDR	Time-domain reflectometry: Counter used to approximate distance to a cable fault. This value is associated with the last occurrence of more than 16 retries.	PD (5,0)	168
ETMRBE	Receive buffer errors: A silo overflow occurred upon receiving a frame.	PD (5,0)	171
ETMSPI	Spurious interrupts: An interrupt was received but could not be decoded into a recognizable interrupt.	PD (5,0)	174
ETMDIF	Discarded inbound frames: Receiver discarded frame due to lack of AIF entries.	PD (5,0)	177
ETMROV	Receive overruns: Receiver has lost all or part of an incoming frame due to buffer shortage.	PD (5,0)	180

Figure A-13 (Page 3 of 3). Ethernet Counter

Field Name	Description	Attributes	Buffer Position
ETMMEE	Memory error: The chipset on the IO adapters is the bus master and did not receive ready signal within 25.6 usecs of asserting the address on the DAL lines.	PD (5,0)	183
ETMIOV	Interrupt overrun: Interrupt not processed due to lack of status queue entries.	PD (5,0)	186
ETMTUN	Transmit underflow: Transmitter has truncated a message due to data late from memory.	PD (5,0)	189
ETMBBE	Babble errors: Transmitter exceeded maximum allowable time on channel.	PD (5,0)	192
ETMSQE	Signal quality error: Signal indicating the transmit is successfully complete did not arrive within 2 usecs of successful transmission.	PD (5,0)	195
ETMM1R	More than 1 retry to transmit: Frame required more than one retry for successful transmission.	PD (5,0)	198
ETM1R	Exactly one retry to transmit: Frame required 1 retry for successful transmission.	PD (5,0)	201
ETMDCN	Deferred conditions: The chipset on the IO adapters deferred transmission due to busy channel.	PD (5,0)	204
ETIOPB	IOP bus number.	PD (3,0)	207
ETIOPA	IOP bus address.	PD (3,0)	209

File Name: QAPMSTNE

Ethernet Station File Entries: Figure A-14 lists the fields in the Ethernet station file.

Figure A-14 (Page 1 of 2). Ethernet Station Counter

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
STIOPI	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
STPCEP	The provider connection end point (PCEP) ID.	C (8)	21
STLND	Line description: The name of the description for this line.	C (10)	29
STSTNN	Station Name: The name of the station on this line.	C (10)	39
STLSPD	Line speed: The line speed expressed in bits per second (bps).	PD (11,0)	49
STTXMT	Total number of Type II frames transmitted.	PD (11,0)	55

Figure A-14 (Page 2 of 2). Ethernet Station Counter

Field Name	Description	Attributes	Buffer Position
STTRCV	Total number of Type II frames received.	PD (11,0)	61
STBXMT	Total number of bytes transmitted in all I-frames.	PD (11,0)	67
STBRCV	Total number of bytes received in all I-frames.	PD (11,0)	73
STIXMT	Total number of I-frames transmitted.	PD (11,0)	79
STIRCV	Total number of I-frames received.	PD (11,0)	85
STIREX	Number of I-frames retransmitted.	PD (11,0)	91
STBREX	Number of bytes retransmitted in I-frames.	PD (11,0)	97
STRNRX	Number of receive-not-ready frames transmitted.	PD (5,0)	103
STRNRR	Number of receive-not-ready frames received.	PD (5,0)	106
STFRMX	Number of frame-reject frames transmitted.	PD (5,0)	109
STFRMR	Number of frame-reject frames received.	PD (5,0)	112
STREJR	Number of reject frames received.	PD (5,0)	115
STREJT	Number of reject frames transmitted.	PD (5,0)	118
STSABX	Number of set asynchronous balanced mode extended frames transmitted.	PD (5,0)	121
STSABR	Number of set asynchronous balanced mode extended frames received.	PD (5,0)	124
STDISX	Number of disconnect frames transmitted.	PD (5,0)	127
STDISR	Number of disconnect frames received.	PD (5,0)	130
STDMFX	Number of disconnect mode frames transmitted.	PD (5,0)	133
STDMFR	Number of disconnect mode frames received.	PD (5,0)	136
STN2RE	N2 retries end count: This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	139
STT1TE	T1 timer end count: Number of times the T1 timer ended. This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	142
STTITE	Ti timer end count: Number of times the Ti timer (inactivity timer) ended.	PD (5,0)	145
STLBCT	Local busy count: Number of times station entered local busy substate.	PD (5,0)	148
STPRCL	Protocol type: T for Ethernet network.	C (1)	151
STIOPB	IOP bus number.	PD (3,0)	152
STIOPA	IOP bus address.	PD (3,0)	154

File Name: QAPMDDI:

QAPMDDI is a database file for distributed data interface (DDI) counter.

Figure A-15 (Page 1 of 2). DDI Counter

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The Nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) for the job interval entry and job completion date, and time (hh:mm:ss) for the job completion entry.	C (12)	4
INTSEC	Elapsed interval seconds.	PD (7,0)	16
DDIOPI	IOP Address	C (1)	20
DDLND	Line description: The name of the description for this line.	C (10)	21
	Reserved.	C (10)	31
DDLSP	Line speed: The line speed expressed in bits per second (bps).	PD (11,0)	41
DLTFT	Total number of Type II frames transmitted.	P (11,0)	47
DLTFR	Total number of Type II frames received.	P (11,0)	53
DLIFT	Total number of characters transmitted in all I-frames.	P (11,0)	59
DLIFR	Total number of characters received in all I-frames.	P (11,0)	65
DLICT	Total number of I-frames characters transmitted.	(11,0)	71
DLICR	Total number of I-frames characters received.	(11,0)	77
DLPRCL	Protocol ID	C (1)	83
DLRFT	Total number of receive-not-ready frames transmitted.	PD (11,0)	84
DLRFR	Total number of receive-not-ready frames received.	PD (11,0)	90
DLFFT	Total number of frame-reject (FRMR) frames transmitted.	PD (11,0)	96
DLFFR	Total number of frame-reject (FRMR) frames received.	PD (11,0)	102
DLRJFR	Number of reject frames received.	PD (11,0)	108
DLRJFT	Number of reject frames transmitted.	PD (11,0)	114
DLSFT	Number of set asynchronous balanced mode extended frames transmitted.	PD (11,0)	120
DLSFR	Number of set asynchronous balanced mode extended frames received.	PD (11,0)	126
DLDFT	Number of disconnect (DISC) frames transmitted.	PD (11,0)	132
DLDFR	Number of disconnect (DISC) frames received.	PD (11,0)	138
DLDMT	Number of disconnect mode (DM) frames transmitted.	PD (11,0)	144
DLDMR	Number of disconnect mode (DM) frames received.	PD (11,0)	150
DLN2R	N2 retries end count: This count is updated when host has attempted to contact a station n times, and the T1 timer ended n times before the station responded.	PD (11,0)	156
DLT1T	T1 timer end count: Number of times the T1 ended. This count is updated when the host has attempted to contact a station n times, and the T1 timer ended n times before the station responded.	PD (11,0)	162
DMFRV	Number of MAC frames received.	PD (11,0)	168
DMFCC	Number of MAC frames copied.	PD (11,0)	174
DMFTR	Number of MAC frames transmitted.	PD (11,0)	180
DMTKN	Number of MAC tokens received.	PD (11,0)	186

Figure A-15 (Page 2 of 2). DDI Counter

Field Name	Description	Attributes	Buffer Position
DMERR	MAC error count.	PD (11,0)	192
DMLFC	Lost frame count.	PD (11,0)	198
DMTVX	TVX expiration count.	PD (11,0)	204
DMNCC	Not copied count.	PD (11,0)	210
DMLAT	MAC late count.	PD (11,0)	216
DLXROP	Ring operation count.	PD (11,0)	222
DMABE	PortA elasticity buffer (EB) errors.	PD (11,0)	228
DMATF	PortA LCT count: count of consecutive times the confidence test (LCT) has failed.	PD (11,0)	234
DMALR	PortA reject count.	PD (11,0)	240
DMAEC	PortA link error monitor (LEM) count.	PD (11,0)	246
DMBBE	PortB elasticity buffer (EB) errors.	PD (11,0)	252
DMBTF	PortB LCT count: count of consecutive times the confidence test (LCT) has failed.	PD (11,0)	258
DMBLR	PortB reject count.	PD (11,0)	264
DMBEC	PortB link error monitor (LEM) count.	PD (11,0)	270
DMANR	Address not recognized.	PD (11,0)	276
DMFNC	Frame not copied.	PD (11,0)	282
DMTKE	Token error count.	PD (11,0)	288
DMDUP	Duplicate address count.	PD (11,0)	294
DMDFR	Discarded frame count.	PD (11,0)	300
DMTXU	Transmit underruns.	PD (11,0)	306
DMRER	Recoverable errors.	PD (11,0)	312
DMNER	Nonrecoverable errors.	PD (11,0)	318
DMSIN	Spurious interruptions.	PD (11,0)	324
DIOPB	IOP bus number.	PD (3,0)	330
DIOPA	IOP bus address.	PD (3,0)	332

File Name: QAPMSTND

Frame Relay Entries: This is the station counter file for distributed data interface (DDI) information. Figure A-16 lists the fields in the DDI station counter. file.

Figure A-16 (Page 1 of 3). DDI Station Counter

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4

Figure A-16 (Page 2 of 3). DDI Station Counter

Field Name	Description	Attributes	Buffer Position
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
SDIOPI	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
SDPCEP	The provider connection end point (PCEP) ID.	C (8)	21
SDLND	Line description: The name of the description for this line.	C (10)	29
SDSTNN	Station name: The name of the station on this line.	C (10)	39
SDLSPD	Line speed: The line speed expressed in bits per second (bps).	PD (11,0)	49
SDTXMT	Total number of Type II frames transmitted.	PD (11,0)	55
SDTRCV	Total number of Type II frames received.	PD (11,0)	61
SDBXMT	Total number of bytes transmitted in all I-frames.	PD (11,0)	67
SDBRCV	Total number of bytes received in all I-frames.	PD (11,0)	73
SDIXMT	Total number of I-frames transmitted.	PD (11,0)	79
SDIRCV	Total number of I-frames received.	PD (11,0)	85
SDIREX	Number of I-frames retransmitted.	PD (11,0)	91
SDBREX	Number of bytes retransmitted in I-frames.	PD (11,0)	97
SDRNRX	Number of receive-not-ready frames transmitted.	PD (5,0)	103
SDRNRR	Number of receive-not-ready frames received.	PD (5,0)	106
SDFRMX	Number of frame-reject frames transmitted.	PD (5,0)	109
SDFRMR	Number of frame-reject frames received.	PD (5,0)	112
SDREJR	Number of reject frames received.	PD (5,0)	115
SDREJT	Number of reject frames transmitted.	PD (5,0)	118
SDSABX	Number of set asynchronous balanced mode extended frames transmitted.	PD (5,0)	121
SDSABR	Number of set asynchronous balanced mode extended frames received.	PD (5,0)	124
SDDISX	Number of disconnect frames transmitted.	PD (5,0)	127
SDDISR	Number of disconnect frames received.	PD (5,0)	130
SDDMFX	Number of disconnect mode frames transmitted.	PD (5,0)	133
SDDMFR	Number of disconnect mode frames received.	PD (5,0)	136
SDN2RE	N2 retries end count: This count is updated when the host has attempted to contact a station *n* times, and the T1 timer ended *n* times before the station responded.	PD (5,0)	139
SDT1TE	T1 timer end count: Number of times the T1 timer ended. This count is updated when the host has attempted to contact a station *n* times, and the T1 timer ended *n* times before the station responded.	PD (5,0)	142

Figure A-16 (Page 3 of 3). DDI Station Counter

Field Name	Description	Attributes	Buffer Position
SDTITE	Ti timer end count: Number of times the Ti timer (inactivity timer) ended.	PD (5,0)	145
SDLBCT	Local busy count: Number of times station entered local busy sub-state.	PD (5,0)	148
SDPRCL	Protocol type: E for token-ring network.	C (1)	151
SDIOPB	IOP bus number.	PD (3,0)	152
SDIOPA	IOP bus address.	PD (3,0)	154

File Name: QAPMFRLY:

QAPMFRLY is a database file for frame relay counter.

Figure A-17 (Page 1 of 2). Frame Relay Counter

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The Nth sample interval since the start of the performance monitor job.	PD (5 0)	1
DTETIM	Interval date (yy/mm/dd) for the job interval entry and job completion date, and time (hh:mm:ss) for the job completion entry.	C (12)	4
INTSEC	Elapsed interval seconds.	PD (7 0)	16
YIOPI	IOP address.	C (1)	20
YLND	Line description: The name of the description for this line.	C (10)	21
YCUD	Reserved.	C (10)	31
YLSP	Line speed: The line speed expressed in bits per second (bps).	PD (11,0)	41
YLTFT	Total number of frames transmitted.	PD (11,0)	47
YLTFR	Total number of frames received.	PD (11,0)	53
YLIFT	Total number of I-frames transmitted.	PD (11,0)	59
YLIFR	Total number of I-frames received.	PD (11,0)	65
YLICT	Total number of I-frames characters transmitted.	PD (11,0)	71
YLICR	Total number of I-frames characters received.	PD (11,0)	77
YLPRCL	Protocol field text.	C (1)	83
YLRFT	Number of receive-not-ready (RNR) frames transmitted.	PD (11,0)	84
YLRFR	Number of receive-not-ready (RNR) frames received.	PD (11,0)	90
YLFFT	Number of frame-reject frames transmitted.	PD (11,0)	96
YLFFR	Total number of frame-reject frames received.	PD (11,0)	102
YLRJFR	Number of reject frames received.	PD (11,0)	108
YLRJFT	Number of reject frames transmitted.	PD (11,0)	114
YLSFT	Number of set asynchronous balanced mode extended (SABME) frames transmitted.	PD (11,0)	120

Figure A-17 (Page 2 of 2). Frame Relay Counter

Field Name	Description	Attributes	Buffer Position
YLSFR	Number of set asynchronous balanced mode extended (SABME) frames received.	PD (11,0)	126
YLDFT	Number of disconnect (DISC) frames transmitted.	PD (11,0)	132
YLDFR	Number of disconnect (DISC) frames received.	PD (11,0)	138
YLDMT	Number of disconnect mode (DM) frames transmitted.	PD (11,0)	144
YLDMR	Number of disconnect mode (DM) frames received.	PD (11,0)	150
YLN2R	N2 retries end count: This count is updated when the host has attempted has attempted to contact a station n times, and the T1 timer ended n times before the station responded.	PD (11,0)	156
YLT1T	T1 timer end count: Number of times the T1 timer ended. This count is updated when the host has attempted to contact a station n times, and the T1 timer ended n times before the station responded.	PD (11,0)	162
YMLTI	Frames transmitted.	PD (11,0)	168
YMLSE	Local management interface (LMI) sequence errors.	PD (11,0)	174
YMLPE	Local management interface (LMI) protocol errors.	PD (11,0)	180
YMPDE	Port monitor data set ready (DSR) errors.	PD (11,0)	186
YMPCE	Port monitor clear to send (CTS) errors.	PD (11,0)	192
YMMER	Address not recognized.	PD (11,0)	198
YIOPB	IOP bus number.	PD (3,0)	204
YIOPA	IOP bus address.	PD (3,0)	206

File Name: QAPMSTNY

FDDI File Entries: Figure A-18 lists the fields in the frame relay station counter.

Figure A-18 (Page 1 of 2). Frame Relay Station Counter

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
SYIOPI	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
SYPCEP	The provider connection end point (PCEP) ID.	C (8)	21
SYLND	Line description: The name of the description for this line.	C (10)	29
SYSTNN	Station name: The name of the station on this line.	C (10)	39

Figure A-18 (Page 2 of 2). Frame Relay Station Counter

Field Name	Description	Attributes	Buffer Position
SYLSPD	Line speed: The line speed expressed in bits per second (bps).	PD (11,0)	49
SYTXMT	Total number of Type II frames transmitted.	PD (11,0)	55
SYTRCV	Total number of Type II frames received.	PD (11,0)	61
SYBXMT	Total number of bytes transmitted in all I-frames.	PD (11,0)	67
SYBRCV	Total number of bytes received in all I-frames.	PD (11,0)	73
SYIXMT	Total number of I-frames transmitted.	PD (11,0)	79
SYIRCV	Total number of I-frames received.	PD (11,0)	85
SYIREX	Number of I-frames retransmitted.	PD (11,0)	91
SYBREX	Number of bytes retransmitted in I-frames.	PD (11,0)	97
SYRNRX	Number of receive-not-ready frames transmitted.	PD (5,0)	103
SYRNRR	Number of receive-not-ready frames received.	PD (5,0)	106
SYFRMX	Number of frame-reject frames transmitted.	PD (5,0)	109
SYFRMR	Number of frame-reject frames received.	PD (5,0)	112
SYREJR	Number of reject frames received.	PD (5,0)	115
SYREJT	Number of reject frames transmitted.	PD (5,0)	118
SYSABX	Number of set asynchronous balanced mode extended frames transmitted.	PD (5,0)	121
SYSABR	Number of set asynchronous balanced mode extended frames received.	PD (5,0)	124
SYDISX	Number of disconnect frames transmitted.	PD (5,0)	127
SYDISR	Number of disconnect frames received.	PD (5,0)	130
SYDMFX	Number of disconnect mode frames transmitted.	PD (5,0)	133
SYDMFR	Number of disconnect mode frames received.	PD (5,0)	136
SYN2RE	N2 retries end count: This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	139
SYT1TE	T1 timer end count: Number of times the T1 timer ended. This count is updated when the host has attempted to contact a station n times and n times the T1 timer ended before the station responded.	PD (5,0)	142
SYTITE	Ti timer end count: Number of times the Ti timer (inactivity timer) ended.	PD (5,0)	145
SYLBCT	Local busy count: Number of times station entered local busy substate.	PD (5,0)	148
SYPRCL	Protocol type: E for token-ring network.	C (1)	151
SYIOPB	IOP bus number.	PD (3,0)	152
SYIOPA	IOP bus address.	PD (3,0)	154

File Name: QAPMSAP

TRLAN, Ethernet, DDI, and Frame Relay SAP File Entries: Figure A-19 lists the fields in the Service Access Point (SAP) file.

Figure A-19. SAP Statistics (One Per SAP Per Line)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
SCIOPI	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
SCSSAP	SSAP ID: The Source SAP (SSAP) ID.	C (2)	21
SCLND	Line description: The name of the description for the line containing the SAP listed above.	C (10)	23
SCLSPD	Line speed: The speed of the line in bits per second (bps).	PD (11,0)	33
SCIRCV	UI frames received: Total number of UI frames received at this SSAP.	PD (11,0)	39
SCIXMT	UI frames transmitted: Total number of UI frames transmitted through this SSAP.	PD (11,0)	45
SCBRCV	UI bytes received: Total number of bytes received at this SSAP contained within a UI frame.	PD (11,0)	51
SCBXMT	UI bytes transmitted: Total number of bytes transmitted through this SSAP contained within a UI frame.	PD (11,0)	57
SCIDSC	Number of UI frames received and discarded by this SSAP.	PD (11,0)	63
SCPRCL	Protocol type: T for Ethernet or E for Token-Ring.	C (1)	69
SCIOPB	IOP bus number.	PD (3,0)	70
SCIOPA	IOP bus address.	PD (3,0)	72

File Name: QAPMLAPD

Integrated Services Digital Network LAPD File Entries: Figure A-20 lists the fields in the LAPD file.

Figure A-20 (Page 1 of 4). ISDN LAPD Counter

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16

Figure A-20 (Page 2 of 4). ISDN LAPD Counter

Field Name	Description	Attributes	Buffer Position
LDIOP	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
LDNWI	Network interface: The name of the network interface description.	C (10)	21
	Reserved	C (10)	31
LDLSP	Link speed: The speed of this channel in bits per second.	PD (11,0)	41
LDPRCL	Protocol type: D for LAPD.	C (1)	47
LPLOFA	Loss of frame alignment: Total number of times when a time period equivalent to two 48-bit frames has elapsed without having detected valid pairs of line code violations.	PD (11,0)	48
LPLECV	Local end code violation: This condition is counted by the terminal equipment (TE) to indicate unintended code violation detected by the TE for frames received on the T interface.	PD (11,0)	54
LPDTSI	Detected access transmission error in (DTSE-in): The number of times the TE received an indication from the network termination type 1 (NT1) that a cycle redundancy check (CRC) error has been detected by the NT1 across the NT1/LT or U interface.	PD (11,0)	60
LPDTSO	Detected access transmission error out (DTSE-out): The number of times the TE received an indication from the NT1 that a CRC error has been detected by the line transition termination (LT) across the U interface.	PD (11,0)	66
LPFECV	Far end code violation: This condition is counted by the TE to indicate unintended code violation detected by the NT1 for frames transmitted to the NT1 on the T interface.	PD (11,0)	72
LPES	Errored seconds: Total number of seconds that had at least one DTSE-in or DTSE-out error.	PD (5,0)	78
LPSES	Severely erred seconds: Total number of seconds that had more than three DTSE-in and DTSE-out errors.	PD (5,0)	81
LPCOL	Collision detect: The number of times the TE detected that its transmitted frame had been corrupted by another TE attempting to use the same bus.	PD (11,0)	84
LLCRCE	Receive CRC errors: The number of received frames that contain a CRC (cycle redundancy check) error.	PD (11,0)	90
LLSFE	Short frame errors: The number of short frames received. A short frame is a frame that has fewer octets between its start flag and end flag than is permitted.	PD (11,0)	96
LLORUN	Receive overrun: The number of times the ISDN subsystem could not keep pace with incoming data because of local controller overload.	PD (11,0)	102
LLURUN	Transmit underrun: The number of times the ISDN subsystem could not keep pace with outgoing data because of local controller overload.	PD (11,0)	108
LLABRT	Aborts received: The number of frames received that contained HDLC abort indicators.	PD (11,0)	114

Figure A-20 (Page 3 of 4). ISDN LAPD Counter

Field Name	Description	Attributes	Buffer Position
LLFRIE	Frames received in error: The sum of receive cycle redundancy check (CRC) errors, short frame errors, receive overrun, transmit underrun, aborts received, and frame sequence errors (LLCRCE, LLSFE, LLORUN, LLURUN, LLABRT, LSSEQE).	PD (11,0)	120
	Reserved.	PD (11,0)	126
	Reserved.	PD (11,0)	132
	Reserved.	PD (11,0)	138
LSFRT	Retransmitted frames.	PD (11,0)	144
LSSEQE	Sequence errors: The number of received frames that contained sequence numbers indicating frames were lost.	PD (11,0)	150
LSFTRN	Total number of frames transmitted: This includes information (I), unnumbered information (UI), and supervisory (S) frames sent to a remote link station. This includes frames retransmitted and frames sent on transmissions stopped by transmit underrun, in addition to successful transmissions.	PD (11,0)	156
LSFRCV	Total number of frames received: This includes information (I), unnumbered information (UI), and supervisory (S) frames received from the remote link station. This includes no errors.	PD (11,0)	162
LSBTRN	Total bytes transmitted: The total number of bytes transmitted to a remote link station. This includes bytes retransmitted and bytes sent on transmissions stopped by a transmit underrun, in addition to successful transmissions.	PD (11,0)	168
LSBRCV	Total bytes received: The total number of bytes received from the remote link station. This includes no errors.	PD (11,0)	174
	Reserved.	PD (11,0)	180
LQTOC	Total outgoing calls: The number of outgoing call attempts. For X.31 this includes outgoing SETUP messages requesting a packet switched connection. For Q.932, outgoing REGISTER messages are not included in this count.	PD (11,0)	186
LQROC	Retry for outgoing calls: The number of outgoing calls that were rejected by the network. For X.31 this includes retry for outgoing SETUP messages requesting a packet switched connection. For Q.932, retry for outgoing REGISTER messages are not included in this count.	PD (11,0)	192
LQTIC	Total incoming calls: The number of incoming call attempts. For X.31 this includes incoming SETUP messages requesting a packet switched connection. For Q.932, incoming REGISTER messages are not included in this count.	PD (11,0)	198
LQRIC	Rejected incoming calls: The number of incoming calls that are rejected by the TE. For passive bus, the call may be intended for another TE that shares the same passive bus. This includes calls rejected both directly by the IOP and by the IOM. For X.31 this includes rejected incoming SETUP messages requesting a packet switched connection. For Q.932, rejected incoming REGISTER messages are not included in this count.	PD (11,0)	204
	Reserved.	PD (11,0)	210
	Reserved.	PD (11,0)	216

Figure A-20 (Page 4 of 4). ISDN LAPD Counter

Field Name	Description	Attributes	Buffer Position
LDIOPB	IOP bus number.	PD (3,0)	222
LDIOPA	IOP bus address.	PD (3,0)	224
LDCHLS1	S1 maintenance channel: Set to one if the S1 maintenance channel was active.	PD (1,0)	226

File Name: QAPMIDLC

Integrated Services Digital Network (ISDN) Data Link Control File Entries: Figure A-21 lists the fields in the ISDN Data Link Control (IDLC) file.

Figure A-21 (Page 1 of 2). ISDN IDLC Counter

Field Name	Description	Attri-butes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
ISIOP	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
ISLND	Line description: The name of the line description.	C (10)	21
ISNWI	Network interface description: The name of the network interface description.	C (10)	31
	Reserved.	C (10)	41
ISLSP	Link speed: The speed of this channel in bits per second.	PD (11,0)	51
ISPRCL	Protocol type: I for IDLC.	C (1)	57
ILCRCE	Receive CRC errors: The number of received frames that contain a cycle redundancy check (CRC) error.	PD (11,0)	58
ILSFE	Short frame errors: The number of short frames received. A short frame is a frame that has fewer octets between its start flag and end flag than is permitted.	PD (11,0)	64
ILORUN	Receive overrun: The number of times the ISDN subsystem could not keep pace with incoming data because of local controller overload.	PD (11,0)	70
ILURUN	Transmit underrun: The number of times the ISDN subsystem could not keep pace with outgoing data because of local controller overload.	PD (11,0)	76
ILABRT	Aborts received: The number of frames received that contained HDLC abort indicators.	PD (11,0)	82

Figure A-21 (Page 2 of 2). ISDN IDLC Counter

Field Name	Description	Attri-butes	Buffer Position
ILFRIE	Frames received in error: The sum of receive CRC errors, short frame errors, receive overrun, transmit underrun, aborts received, and frame sequence errors (ILCRCE, ILSFE, ILORUN, ILURUN, ILABRT, ISSEQE).	PD (11,0)	88
	Reserved.	PD (11,0)	94
	Reserved.	PD (11,0)	100
	Reserved.	PD (11,0)	106
ISFRT	Retransmitted frames:	PD (11,0)	112
ISSEQE	Sequence errors: The number of received frames that contained sequence numbers indicating frames were lost.	PD (11,0)	118
ISFTRN	Total number of frames transmitted: This includes information (I), unnumbered information (UI), and supervisory (S) frames sent to a remote link station. This includes frames retransmitted and frames sent on transmissions stopped by transmit underruns, in addition to successful transmissions.	PD (11,0)	124
ISFRCV	Total number of frames received: This includes information (I), unnumbered information (UI), and supervisory (S) frames received from the remote link station. This includes no errors.	PD (11,0)	130
ISBTRN	Total bytes transmitted: The total number of bytes transmitted to a remote link station. This includes bytes retransmitted and bytes sent on transmissions stopped by a transmit underrun, in addition to successful transmissions.	PD (11,0)	136
ISBRCV	Total bytes received: The total number of bytes received from the remote link station. This includes no errors.	PD (11,0)	142
	Reserved.	PD (11,0)	148
ISIOPB	IOP bus number.	PD (3,0)	154
ISIOPA	IOP bus address.	PD (3,0)	156
ISB1	B1 channel: Set to one if the B1 channel was used.	PD (1,0)	158
ISB2	B2 channel: Set to one if the B2 channel was used.	PD (1,0)	159

File Name: QAPMBUS

Licensed Internal Code bus counters:
Figure A-22 lists the fields in the bus counters file.

Figure A-22 (Page 1 of 2). Bus Counter

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4

Figure A-22 (Page 2 of 2). Bus Counter

Field Name	Description	Attributes	Buffer Position
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
BUIOPB	IOP bus number.	PD (3,0)	20
BUOPSR	Number of OPSTARTs received: RRCB in server storage.	PD (11,0)	22
BUSGLR	Signals received.	PD (11,0)	28
BUOPSS	Number of OPSTARTs sent.	PD (11,0)	34
BUSGLS	Signals sent.	PD (11,0)	40
BURSTQ	Restart queues sent.	PD (11,0)	46
BUBNAR	Occurrences of BNA received.	PD (11,0)	52

File Name: QAPMCIOP

Communications Controller File Entries:
Figure A-23 lists the fields in the communications controller file.

Figure A-23 (Page 1 of 2). Communications Controller IOP Data (One for Each Communications Controller)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
CIIOP	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
CTIPKT	Total packets transferred.	PD (11,0)	21
CIDMAO	Total bytes transmitted from an IOP to the system across the bus.	PD (11,0)	27
CIDMAI	Total bytes transmitted to the IOP from the system across the bus.	PD (11,0)	33
CIOPSR	OPSTART bus unit message received from another bus unit using normal flow.	PD (11,0)	39
CIOPSS	OPSTART bus unit message received from another bus unit using reverse flow method 2 (always 0).	PD (11,0)	45
CISGLR	Signals received.	PD (11,0)	51
CIOPST	OPSTARTS sent.	PD (11,0)	57
CISLGS	Signals sent.	PD (11,0)	63
CIRSTQ	Restart queues sent.	PD (11,0)	69

Figure A-23 (Page 2 of 2). Communications Controller IOP Data (One for Each Communications Controller)

Field Name	Description	Attributes	Buffer Position
CIRQDO	DMA requests sent for output of data: The number of requests the IOP sends to the system for data to be sent from the IOP to the system across the bus.	PD (11,0)	75
CIRQDI	DMA requests sent for input of data: The number of requests the IOP sends to the system for data to be sent to the IOP from the system across the bus.	PD (11,0)	81
CIBNAR	Occurrences of BNA received.	PD (11,0)	87
CIPRCU	Processor utilization: The number of fixed-time intervals that this communications IOP spent in the idle state.	PD (11,0)	93
CIIDLC	Idle loop count (see note): The number of times the communications IOP ran an idle loop. This is done when the IOP has no work to perform. This count is used with the idle loop time.	PD (11,0)	99
CIIDLT	Idle loop time (see note): The time (in hundredths of microseconds) to run the idle loop once.	PD (11,0)	105
CIRAMU	Available local storage (in bytes): The number of bytes of free local storage in the IOP. The free local storage will probably be non-contiguous because of fragmentation.	PD (11,0)	111
CIIOPB	IOP bus number.	PD (3,0)	117
CIIOPA	IOP bus address.	PD (3,0)	119
CITYPE	The type of IOP described by this record.	C (4)	121
CISYSF	The total time (in milliseconds) used by this IOP for basic system function.	PD (11,0)	125
CICOMM	Combined time (in milliseconds) accounted for by all protocols.	PD (11,0)	131
CISDLC	Time (in milliseconds) accounted for by SDLC protocols.	PD (11,0)	137
CIASYN	Time (in milliseconds) accounted for by ASYNC protocols.	PD (11,0)	143
CIBSC	Time (in milliseconds) accounted for by ISYNC protocols.	PD (11,0)	149
CIX25L	Time (in milliseconds) accounted for by X.25 LLC.	PD (11,0)	155
CIX25P	Time (in milliseconds) accounted for by X.25 PLC.	PD (11,0)	161
CIX25D	Time (in milliseconds) accounted for by X.25 DLC.	PD (11,0)	167
CILAN	Time (in milliseconds) accounted for by TRLAN.	PD (11,0)	173
CILAP	Time (in milliseconds) accounted for by ISDN LAPD, LAPE, and PMI.	PD (11,0)	179
CIQ931	Time (in milliseconds) accounted for by ISDN Q.931.	PD (11,0)	185

Note: The idle loop count and time are used to calculate the communications IOP utilization as follows:

Convert the product of the idle loop count times the idle loop time from hundredths of microseconds to seconds. Subtract this from the interval time, and divide the result by the interval time. For example:

```
IOP utilization = (INTSEC - (CIIDLC * CIIDLT)/10**8)/ INTSEC
```

File Name: QAPMMIOP

Multifunction Controller File Entries:
Figure A-24 lists the fields in the multifunction controller file.

Figure A-24 (Page 1 of 2). Multifunction Controller IOP Data (One for Each Multifunction Controller)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
MIIOP	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
MIPRCU	Processor utilization: The number of fixed-time intervals that this multifunction IOP spent in the idle state.	PD (11,0)	21
MIRAMU	Available local storage (in bytes): The number of bytes of free local storage in the IOP. The free local storage will probably be noncontiguous because of fragmentation.	PD (11,0)	27
MITPKT	Total packets transferred.	PD (11,0)	33
MIDMAO	Total bytes transmitted from an IOP to the system across the bus.	PD (11,0)	39
MIDMAI	Total bytes transmitted to the IOP from the system across the bus.	PD (11,0)	45
MIOPSR	OPSTART bus unit message received from another bus unit using normal flow.	PD (11,0)	51
MIOPSS	OPSTART bus unit message received from another bus unit using reverse flow method 2 (always 0).	PD (11,0)	57
MISGLR	Signals received.	PD (11,0)	63
MIOPST	OPSTARTS sent.	PD (11,0)	69
MISLGS	Signals sent.	PD (11,0)	75
MIRSTQ	Restart queues sent.	PD (11,0)	81
MIRQDO	DMA requests sent for output of data: The number of requests the IOP sends to the system for data to be sent from the IOP to the system across the bus.	PD (11,0)	87
MIRQDI	DMA requests sent for input of data: The number of requests the IOP sends to the system for data to be sent to the IOP from the system across the bus.	PD (11,0)	93
MIBNAR	Occurrences of BNA received.	PD (11,0)	99
MIIDLC	Idle loop count (see note): The number of times the multifunction IOP ran an idle loop. This is done when the IOP has no work to perform. This count is used with the idle loop time.	PD (11,0)	105
MIIDLT	Idle loop time (see Note): The time (in hundredths of microseconds) to run the idle loop once.	PD (11,0)	111
MISYSF	IOP system function time: Total processing unit time (in milliseconds) used by system function tasks.	PD (11,0)	117
MIDISK	Disk time: Total processing unit time (in milliseconds) used by disk tasks.	PD (11,0)	123

Figure A-24 (Page 2 of 2). Multifunction Controller IOP Data (One for Each Multifunction Controller)

Field Name	Description	Attributes	Buffer Position
	Reserved.	PD (11,0)	129
MICOMM	Total communications time: Total processing unit time (in milliseconds) used by all the communications protocols.	PD (11,0)	135
MISDLC	SDLC communications time: Total processing unit time (in milliseconds) used by SDLC communications tasks.	PD (11,0)	141
MIASYN	ASYNC communications time: Total processing unit time (in milliseconds) used by asynchronous communications tasks.	PD (11,0)	147
MIBSC	BSC communications time: Total processing unit time (in milliseconds) used by BSC communications tasks.	PD (11,0)	153
MIX25L	X.25 LLC communications time: Total processing unit time (in milliseconds) used by X.25 LLC communications tasks.	PD (11,0)	159
MIX25P	X.25 PLC communications time: Total processing unit time (in milliseconds) used by X.25 PLC communications tasks.	PD (11,0)	165
MIX25D	X.25 DLC communications time: Total processing unit time (in milliseconds) used by X.25 DLC communications tasks.	PD (11,0)	171
MILAN	LAN communications time: Total processing unit time (in milliseconds) used by token-ring network and Ethernet communications tasks.	PD (11,0)	177
MISDLD	SDLC communications time: Total processing unit time (in milliseconds) used by SDLC communications tasks.	PD (11,0)	183
MIRV02	ISDN communications time: Total processing unit time (in milliseconds) used by ISDN LAPD, LAPE, and PMI communications tasks.	PD (11,0)	189
MIRV03	ISDN communications time: Total processing unit time (in milliseconds) used by ISDN Q.931 communications tasks.	PD (11,0)	195
	Reserved.	PD (11,0)	201
	Reserved.	PD (11,0)	207
	Reserved.	PD (11,0)	213
MISP	Service processor time: Total processing unit time (in milliseconds) used by service processor tasks.	PD (11,0)	219
MIIOPB	IOP bus number.	PD (3,0)	225
MIIOPA	IOP bus address.	PD (3,0)	227
MITYPE	IOP type.	C (4)	229

Note: The idle loop count and time are used to calculate the multifunction IOP utilization as follows:

Convert the product of the idle loop count times the idle loop time from hundredths of microseconds to seconds. Subtract this from the interval time, and divide the results by the interval time. For example:

```
IOP utilization = (INTSEC - (MIIDLC * MIIDLT)/10**8)/ INTSEC
```

File Name: QAPMDIOP

Storage Device Controller File Entries:
Figure A-25 lists the fields in the storage device controller file.

Consider the following information when using this table:

- *Device* means disk.
- The idle loop count and time are used to calculate the storage device controller IOP utilization as follows:

  Convert the product of the idle loop count times the idle loop time from hundredths of microseconds to seconds. Subtract this from

the interval time, and divide the result by the interval time. For example:

```
IOP utilization = (INTSEC - (DIIDLC * DIIDLT)/10**8)/ INTSEC
```

Figure A-25 (Page 1 of 3). Storage Device Controller IOP Data (One for Each Controller)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
DIIOP	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
DIIDLC	Idle loop count: The number of times the disk controller IOP ran an idle loop. This is done when the IOP has no work to perform. This count is used with the idle loop time.	PD (11,0)	21
DIIDLT:	Idle loop time: The time (in hundredths of microseconds) to run the idle loop once.	PD (11,0)	27
DIRID0	Device 0 resource name: Device resource name used to refer to this device. The storage devices are attached on these ports, and there are 16 ports. Field cannot be displayed.	C (8)	33
DISMP0	Number of sample of device 0: The number of times during the data collection interval that the device 0 queue was sampled to see if any I/O requests were waiting for service.	PD (11,0)	41
DIQLN0	Device 0 queue length: The number of I/O requests waiting for service on the device 0 queue at sample time.	PD (11,0)	47
DINRQ0	No-request-serviced count: The number of times that no I/O requests were waiting for service when the queue was sampled.	PD (11,0)	53
DIRID1	Device 1 resource name. Field cannot be displayed.	C (8)	59
DISMP1	Number of samples on device 1.	PD (11,0)	67
DIQLN1	Device 1 queue length.	PD (11,0)	73
DINRQ1	No-request-serviced count.	PD (11,0)	79
DIRID2	Device 2 resource name. Field cannot be displayed.	C (8)	85
DISMP2	Number of samples on device 2.	PD (11,0)	93
DIQLN2	Device 2 queue length.	PD (11,0)	99
DINRQ2	No-request-serviced count.	PD (11,0)	105
DIRID3	Device 3 resource name. Field cannot be displayed.	C (8)	111
DISMP3	Number of samples on device 3.	PD (11,0)	119
DIQLN3	Device 3 queue length.	PD (11,0)	125
DINRQ3	No-request-serviced count.	PD (11,0)	131
DIRID4	Device 4 resource name. Field cannot be displayed.	C (8)	137

Figure A-25 (Page 2 of 3). Storage Device Controller IOP Data (One for Each Controller)

Field Name	Description	Attributes	Buffer Position
DISMP4	Number of samples on device 4.	PD (11,0)	145
DIQLN4	Device 4 queue length.	PD (11,0)	151
DINRQ4	No-request-serviced count.	PD (11,0)	157
DIRID5	Device 5 resource name. Field cannot be displayed.	C (8)	163
DISMP5	Number of samples on device 5.	PD (11,0)	171
DIQLN5	Device 5 queue length.	PD (11,0)	177
DINRQ5	No-request-serviced count.	PD (11,0)	183
DIRID6	Device 6 resource name. Field cannot be displayed.	C (8)	189
DISMP6	Number of samples on device 6.	PD (11,0)	197
DIQLN6	Device 6 queue length.	PD (11,0)	203
DINRQ6	No-request-serviced count.	PD (11,0)	209
DIRID7	Device 7 resource name. Field cannot be displayed.	C (8)	215
DISMP7	Number of samples on device 7.	PD (11,0)	223
DIQLN7	Device 7 queue length.	PD (11,0)	229
DINRQ7	No-request-serviced count.	PD (11,0)	235
DIRID8	Device 8 resource name. Field cannot be displayed.	C (8)	241
DISMP8	Number of samples on device 8.	PD (11,0)	249
DIQLN8	Device 8 queue length.	PD (11,0)	255
DINRQ8	No-request-serviced count.	PD (11,0)	261
DIRID9	Device 9 resource name. Field cannot be displayed.	C (8)	267
DISMP9	Number of samples on device 9.	PD (11,0)	275
DIQLN9	Device 9 queue length.	PD (11,0)	281
DINRQ9	No-request-serviced count.	PD (11,0)	287
DIRIDA	Device 10 resource name. Field cannot be displayed.	C (8)	293
DISMPA	Number of samples on device 10.	PD (11,0)	301
DIQLNA	Device 10 queue length.	PD (11,0)	307
DINRQA	No-request-serviced count.	PD (11,0)	313
DIRIDB	Device 11 resource name. Field cannot be displayed.	C (8)	319
DISMPB	Number of samples on device 11.	PD (11,0)	327
DIQLNB	Device 11 queue length.	PD (11,0)	333
DINRQB	No-request-serviced count.	PD (11,0)	339
DIRIDC	Device 12 resource name. Field cannot be displayed.	C (8)	345
DISMPC	Number of samples on device 12.	PD (11,0)	353
DIQLNC	Device 12 queue length.	PD (11,0)	359
DINRQC	No-request-serviced count.	PD (11,0)	365
DIRIDD	Device 13 resource name. Field cannot be displayed.	C (8)	371
DISMPD	Number of samples on device 13.	PD (11,0)	379
DIQLND	Device 13 queue length.	PD (11,0)	385

Figure A-25 (Page 3 of 3). Storage Device Controller IOP Data (One for Each Controller)

Field Name	Description	Attributes	Buffer Position
DINRQD	No-request-serviced count.	PD (11,0)	391
DIRIDE	Device 14 resource name. Field cannot be displayed.	C (8)	397
DISMPE	Number of samples on device 14.	PD (11,0)	405
DIQLNE	Device 14 queue length.	PD (11,0)	411
DINRQE	No-request-serviced count.	PD (11,0)	417
DIRIDF	Device 15 resource name. Field cannot be displayed.	C (8)	423
DISMPF	Number of samples on device 15.	PD (11,0)	431
DIQLNF	Device 15 queue length.	PD (11,0)	437
DINRQF	No-request-serviced count.	PD (11,0)	443
DITPDK	Total packets transferred.	PD (11,0)	449
DIDMAO	Total bytes transmitted from the IOP to the system across the bus.	PD (11,0)	455
DIDMAI	Total bytes transmitted to the IOP from the system across the bus.	PD (11,0)	461
DIOPSR	OPSTART bus unit message received from another bus unit using normal flow.	PD (11,0)	467
DIOPSS	OPSTART bus unit message received from another bus unit using reverse flow method 2 (always 0).	PD (11,0)	473
DISGLR	Signals received.	PD (11,0)	479
DIOPST	OPSTARTS sent.	PD (11,0)	485
DISGLS	Signals sent.	PD (11,0)	491
DIRSTQ	Restart queues sent.	PD (11,0)	497
DIRQDO	DMA requests sent for output of data: The number of requests the IOP sends to the system for data to be sent from the IOP to the system across the bus.	PD (11,0)	503
DIRQDI	DMA requests sent for input of data: The number of requests the IOP sends to the system for data to be sent to the IOP from the system across the bus.	PD (11,0)	509
DIBNAR	Occurrences of BNA received.	PD (11,0)	515
DIIOPB	IOP bus number.	PD (3,0)	521
DIIOPA	IOP bus address.	PD (3,0)	523
DITYPE	IOP type.	C (4)	525

File Name: QAPMLIOP

Twinaxial IOP Data File Entries: Figure A-26 lists the fields in the twinaxial IOP data file.

Figure A-26 (Page 1 of 3). Twinaxial IOP Data (One for Each Work Station Controller)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4

Figure A-26 (Page 2 of 3). Twinaxial IOP Data (One for Each Work Station Controller)

Field Name	Description	Attributes	Buffer Position
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
LIIOP	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
LIRIDC	Resource ID of controller: The resource ID of the IOP. Field cannot be displayed.	C (8)	21
LITPKT	Total packets transferred.	PD (11,0)	29
LIDMAO	Total bytes transmitted from the IOP to the system across the bus.	PD (11,0)	35
LIDMAI	Total bytes transmitted to the IOP from the system across the bus.	PD (11,0)	41
LIOPSR	OPSTART bus unit message received from another bus unit using normal flow.	PD (11,0)	47
LIOPSS	OPSTART bus unit message received from another bus unit using reverse flow method 2.	PD (11,0)	53
LISGLR	Signal bus unit message received from another bus unit.	PD (11,0)	59
LIOPST	OPSTARTS sent to another bus unit using reverse flow method 2.	PD (11,0)	65
LISGLS	Signals sent to another bus unit.	PD (11,0)	71
LIRSTQ	Restart queues bus unit message sent to another bus unit.	PD (11,0)	77
LIRQDO	DMA requests sent for output of data: The number of requests the IOP sends to the system for data to be sent from the IOP to the system across the bus.	PD (11,0)	83
LIRQDI	DMA requests sent for input of data: The number of requests the IOP sends to the system for data to be sent to the IOP from the system across the bus.	PD (11,0)	89
LIBNAR	Occurrences of BNA received.	PD (11,0)	95
LIIOQC	Wait-on-I/O queue count: The number of I/O requests on the wait-on-I/O queue at sample time. The wait-on-I/O queue holds I/O requests that are being processed or waiting to be processed.	PD (11,0)	101
LISQC	Suspend queue count: The number of elements on the suspend queue at sample time.	PD (11,0)	107
LIAQC	Active queue count: The number of elements on the active queue at sample time. The active queue holds I/O requests that were sent from the host system and were not yet sent to the wait-on-I/O queue.	PD (11,0)	113
LITWIU	Twinaxial use count: The number of times when the wait-on-I/O queue was sampled and the count was not zero (I/O in progress). If this value is divided by the sample count, the result (times 100) is the percentage of time when I/O is occurring.	PD (5,0)	119
LISMPL	Sample count: The number of times during the snapshot interval that the various IOP queues were sampled.	PD (5,0)	122

Figure A-26 (Page 3 of 3). Twinaxial IOP Data (One for Each Work Station Controller)

Field Name	Description	Attributes	Buffer Position
LIIDLC	Idle counts (see Note): The number of times the work station IOP ran an idle loop. This is done when the IOP has no work to perform. This count is used with the idle loop time.	PD (11,0)	125
LIIDLT	Idle loop time (times 0.01 microsecond): The time (in hundredths of microseconds) to run the idle loop once (see Note).	PD (11,0)	131
LIIOPB	IOP bus number.	PD (3,0)	137
LIIOPA	IOP bus address.	PD (3,0)	139
LITYPE	IOP type.	C (4)	141

Note: The idle loop count and time are used to calculate the work station IOP utilization as follows:

Convert the product of the idle loop count times the idle loop time from hundredths of microseconds to seconds. Subtract this from the interval time, and divide the result by the interval time. For example:

```
IOP utilization = (INTSEC - (LIIDLC * LIIDLT)/10**8)/ INTSEC
```

File Name: QAPMRESP

Local Work Station Response Time File Entries: Figure A-27 lists the fields in the local work station response time file.

Figure A-27 (Page 1 of 2). Local Work Station Response Time (One for Each Station)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
LRIOP	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 through 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 through 31. Field cannot be displayed.	C (1)	20
LRWSN	Reserved.	C (10)	21
LRBKT1	Transactions in first response time monitor bracket: The number of transactions from 0 up to and including n seconds for this work station during the *snapshot* interval. The n value is the response time monitor 1 bracket upper limit, and is specified when the performance monitor is started by the STRPFRMON command. A transaction is defined as the time from when the keyboard locked because the Enter key or a function key was pressed to the time when the keyboard unlocked because the display was refreshed.	PD (7,0)	31
LRBKT2	Transactions in second response time monitor bracket: The number of transactions greater than the response time monitor 1 and up to and including response time monitor 2 limits.	PD (7,0)	35

Figure A-27 (Page 2 of 2). Local Work Station Response Time (One for Each Station)

Field Name	Description	Attributes	Buffer Position
LRBKT3	Transactions in third response time monitor bracket: The number of transactions greater than the response time monitor 2 and up to and including response time monitor 3 limits.	PD (7,0)	39
LRBKT4	Transactions in fourth response time monitor bracket: The number of transactions greater than the response time monitor 3 and up to and including response time monitor 4 limits.	PD (7,0)	43
LRBKT5	Transactions in fifth response time monitor bracket: The number of transactions above (longer) than the response time monitor 4 limit.	PD (7,0)	47
LRIOPB	IOP bus number.	PD (3,0)	51
LRIOPA	IOP bus address.	PD (3,0)	53
LRPORT	Work station port number.	PD (3,0)	55
LRSTN	Work station number.	PD (3,0)	57
LRWSID	Work station address. Character data cannot be displayed.	C (5)	59
LRTRNS	The total of all the individual times for all exchanges measured and reported by this record including overflows (LRBKT5). The total time in seconds for all transactions.	PD (7,0)	64

File Name: QAPMRWS

Remote Work Station Response Time File Entries: Figure A-28 lists the fields in the remote work station response time file.

Figure A-28 (Page 1 of 2). Remote Work Station Response Time (One for Each Station)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds during which these transactions occurred.	PD (7,0)	16
RWIOP	IOP address: An 8-bit field made up of two subfields: IOP bus number: Bits 0, 1, and 2 define the I/O bus number for this IOP. The bus number can range from 0 to 7. IOP bus address: Bits 3, 4, 5, 6, and 7 define the address of this IOP on this bus. The bus address can range from 0 to 31. Field cannot be displayed.	C (1)	20
RWWSN	Reserved	C (10)	21
RWBKT1	Transactions in first response time monitor bracket: The number of transactions greater than 0 up to and including n seconds for this work station during the *snapshot* interval. The n value is the upper limit for the first response time monitor bracket, and is specified when the performance monitor is started by the STRPFRMON command. A transaction is defined as the time from when the keyboard locked because the Enter key or a function key was pressed to the time the keyboard is unlocked because the display was refreshed.	PD (7,0)	31

Figure A-28 (Page 2 of 2). Remote Work Station Response Time (One for Each Station)

Field Name	Description	Attributes	Buffer Position
RWBKT2	Transactions in second response time monitor bracket: The number of transactions greater than the response time monitor 1 and up to and including response time monitor 2 limits.	PD (7,0)	35
RWBKT3	Transactions in third response time monitor bracket: The number of transactions greater than the response time monitor 2 and up to and including the response time monitor 3 limits.	PD (7,0)	39
RWBKT4	Transactions in fourth response time monitor bracket: The number of transactions greater than the response time monitor 3 and up to and including the response time monitor 4 limits.	PD (7,0)	43
RWBKT5	Transactions in fifth response time monitor bracket: The number of transactions longer than the limit for the response time monitor 4.	PD (7,0)	47
RWTRNS	The total of all the individual times for all exchanges measured and reported by this record including overflows (RWBKT5). The total time in seconds for all transactions.	PD (7,0)	51
RWIOPB	IOP bus number.	PD (3,0)	55
RWIOPA	IOP bus address.	PD (3,0)	57
RWPORT	Work station port number.	PD (3,0)	59
RWSTN	Work station number for this port.	PD (3,0)	61
RWCUD	Controller description: The name of the controller this work station is attached to.	C (10)	63
RWWSID	Work station address: Character data cannot be displayed.	C (5)	73
RWLND	Line description: Name of the communications line this work station and its controller are attached to.	C (10)	78

File Name: QAPMSNA

SNA data: Figure A-29 lists the fields in the systems network architecture (SNA) file.

Figure A-29 (Page 1 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
SCTLNM	Controller description name.	C (10)	20
SLINNM	Line description name.	C (10)	30
STSKNM	T2 station I/O manager (SIOM) task name.	C (6)	40
SLIOMT	Line I/O manager task name.	C (6)	46
SACPNM	Adjacent control point (CP) name.	C (8)	52
SANWID	Adjacent network ID.	C (8)	60
SAPPN	APPN-capable (Y=yes, N=no).	C (1)	68

Figure A-29 (Page 2 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
SCTYP	Controller type (A=APPC, H=Host).	C (1)	69
SSMFS	Send maximum frame size.	PD (11,0)	70
SRMFS	Receive maximum frame size.	PD (11,0)	76
STLLBU	Date (yy/mm/dd) and time(hh:mm:ss) when most recent connection was established with the adjacent system.	C (12)	82
SNLBU	Number of times a connection has been established with the remote system.	PD (11,0)	94
STACVO	Cumulative elapsed time for automatically created and/or varied-on devices.	PD (11,0)	100
SNACVO	Number of automatically created and/or varied-on devices.	PD (11,0)	106
SNADD	Number of automatically deleted devices.	PD (11,0)	112
SNWAIN	Number of work activities coming in from other T2 SIOM tasks (for example, messages received).	PD (11,0)	118
SNWAOU	Number of work activities sent out to other T2 SIOM tasks (for example, messages received).	PD (11,0)	124
	The following fields refer to end point sessions attributes.		
ENNSS	Number of network priority sessions started.	PD (11,0)	130
ENNSE	Number of network priority sessions ended.	PD (11,0)	136
ENNBB	Number of request units with begin bracket sent and received for all network priority sessions.	PD (11,0)	142
ENNEB	Number of request units with end bracket sent and received for all network priority sessions.	PD (11,0)	148
ENSPWT	The cumulative wait time for all network priority sessions (in milliseconds) caused by session-level send messages. This wait time measures the amount of time application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system.	PD (11,0)	154
ENSPNW	Number of waits occurring for all network priority sessions for session-level send pacing. That is, the number of times application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system.	PD (11,0)	160
ENSPPW	Number of potential waits occurring for all network priority sessions for session-level send pacing. This is the worst case that would occur if the sending of application data was delayed waiting for every pacing response sent by the adjacent system.	PD (11,0)	166
ENSPWS	The cumulative window size for all network priority sessions for session-level send pacing. Each time a pacing response is received from the adjacent system on a network priority session, this count is increased by window size specified by the pacing response.	PD (11,0)	172
ENIPWT	The cumulative wait time for all network priority sessions (in milliseconds) for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	178

Figure A-29 (Page 3 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
ENIPNW	Number of waits occurring for all network priority sessions for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	184
ENQNRE	Number of network priority request/response units entering the transmission priority queue.	PD (11,0)	190
ENQLRE	Length of network priority request/response units entering the transmission priority queue.	PD (11,0)	196
ENQNRL	Number of network priority request/response units leaving the transmission priority queue.	PD (11,0)	177
ENQLRL	Length of network priority request/response units leaving the transmission priority queue.	PD (11,0)	208
ENQTRR	Cumulative wait time in network transmission priority queue.	PD (11,0)	214
ENNRUD	Number of network priority request/response units delivered to the adjacent system.	PD (11,0)	220
ENLRUD	Length of network priority request/response units delivered to the adjacent system.	PD (11,0)	226
ENTRUD	Cumulative service time to deliver a network priority request/response unit to the adjacent system.	PD (11,0)	238
ENNRUR	Number of network priority request/response units received from the adjacent system.	PD (11,0)	232
ENLRUR	Length of network priority request/response units received from the adjacent system.	PD (11,0)	244
EHNSS	Number of high priority sessions started	PD (11,0)	250
EHNSE	Number of high priority sessions ended	PD (11,0)	256
EHNBB	Number of request units with begin bracket sent and received for all high priority sessions	PD (11,0)	262
EHNEB	Number of request units with end bracket sent and received for all high priority sessions	PD (11,0)	268
EHSPWT	The cumulative wait time for all high priority sessions (in milliseconds) caused by session-level send messages. This wait time measures the amount of time application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system.	PD (11,0)	274
EHSPNW	Number of waits occurring for all high priority sessions for session-level send pacing. That is, the number of times application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system.	PD (11,0)	280
EHSPPW	Number of potential waits occurring for all high priority sessions for session-level send pacing. This is the worst case that would occur if the sending of application data was delayed waiting for every pacing response sent by the adjacent system.	PD (11,0)	286
EHSPWS	The cumulative window size for all high priority sessions for session-level send pacing. Each time a pacing response is received from the adjacent system on a network priority session, this count is increased by window size specified by the pacing response.	PD (11,0)	292

Figure A-29 (Page 4 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
EHIPWT	The cumulative wait time for all high priority sessions (in milliseconds) for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	298
EHIPNW	Number of waits occurring for all high priority sessions for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	304
EHQNRE	Number of high priority request/response units entering the transmission priority queue.	PD (11,0)	310
EHQLRE	Length of high priority request/response units entering the transmission priority queue.	PD (11,0)	316
EHQNRL	Number of high priority request/response units leaving the transmission priority queue.	PD (11,0)	322
EHQLRL	Length of high priority request/response units leaving the transmission priority queue.	PD (11,0)	328
EHQTRR	Cumulative wait time in high transmission priority queue.	PD (11,0)	334
EHNRUD	Number of high priority request/response units delivered to the adjacent system.	PD (11,0)	340
EHLRUD	Length of high priority request/response units delivered to the adjacent system.	PD (11,0)	346
EHTRUD	Cumulative service time to deliver a high priority request/response unit to the adjacent system.	PD (11,0)	352
EHNRUR	Number of high priority request/response units received from the adjacent system.	PD (11,0)	358
EHLRUR	Length of high priority request/response units received from the adjacent system.	PD (11,0)	364
EMNSS	Number of medium priority sessions started	PD (11,0)	370
EMNSE	Number of medium priority sessions ended	PD (11,0)	376
EMNBB	Number of request units with begin bracket sent and received for all medium priority sessions	PD (11,0)	382
EMNEB	Number of request units with end bracket sent and received for all medium priority sessions	PD (11,0)	388
EMSPWT	The cumulative wait time for all medium priority sessions (in milliseconds) caused by session-level send messages. This wait time measures the amount of time application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system	PD (11,0)	394
EMSPNW	Number of waits occurring for all medium priority sessions for session-level send pacing. That is, the number of times application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system	PD (11,0)	400
EMSPPW	Number of potential waits occurring for all medium priority sessions for session-level send pacing. This is the worst case that would occur if the sending of application data was delayed waiting for every pacing response sent by the adjacent system.	PD (11,0)	406

Figure A-29 (Page 5 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
EMSPWS	The cumulative window size for all medium priority sessions for session-level send pacing. Each time a pacing response is received from the adjacent system on a network priority session, this count is increased by window size specified by the pacing response	PD (11,0)	412
EMIPWT	The cumulative wait time for all medium priority sessions (in milliseconds) for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	418
EMIPNW	Number of waits occurring for all medium priority sessions for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	424
EMQNRE	Number of medium priority request/response units entering the transmission priority queue.	PD (11,0)	430
EMQLRE	Length of medium priority request/response units entering the transmission priority queue.	PD (11,0)	436
EMQNRL	Number of medium priority request/response units leaving the transmission priority queue.	PD (11,0)	442
EMQLRL	Length of medium priority request/response units leaving the transmission priority queue.	PD (11,0)	448
EMQTRR	Cumulative wait time in medium transmission priority queue.	PD (11,0)	454
EMNRUD	Number of medium priority request/response units delivered to the adjacent system.	PD (11,0)	460
EMLRUD	Length of medium priority request/response units delivered to the adjacent system.	PD (11,0)	466
EMTRUD	Cumulative service time to deliver a medium priority request/response unit to the adjacent system.	PD (11,0)	472
EMNRUR	Number of medium priority request/response units received from the adjacent system.	PD (11,0)	478
EMLRUR	Length of medium priority request/response units received from the adjacent system.	PD (11,0)	484
ELNSS	Number of low priority sessions started	PD (11,0)	490
ELNSE	Number of low priority sessions ended	PD (11,0)	496
ELNBB	Number of request units with begin bracket sent and received for all low priority sessions	PD (11,0)	502
ELNEB	Number of request units with end bracket sent and received for all low priority sessions	PD (11,0)	508
ELSPWT	The cumulative wait time for all low priority sessions (in milliseconds) caused by session-level send messages. This wait time measures the amount of time application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system	PD (11,0)	514
ELSPNW	Number of waits occurring for all low priority sessions for session-level send pacing. That is, the number of times application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system	PD (11,0)	520

Figure A-29 (Page 6 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
ELSPPW	Number of potential waits occurring for all low priority sessions for session-level send pacing. This is the worst case that would occur if the sending of application data was delayed waiting for every pacing response sent by the adjacent system.	PD (11,0)	526
ELSPWS	The cumulative window size for all low priority sessions for session-level send pacing. Each time a pacing response is received from the adjacent system on a network priority session, this count is increased by window size specified by the pacing response	PD (11,0)	532
ELIPWT	The cumulative wait time for all low priority sessions (in milliseconds) for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	538
ELIPNW	Number of waits occurring for all low priority sessions for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	544
ELQNRE	Number of low priority request/response units entering the transmission priority queue.	PD (11,0)	550
ELQLRE	Length of low priority request/response units entering the transmission priority queue.	PD (11,0)	556
ELQNRL	Number of low priority request/response units leaving the transmission priority queue.	PD (11,0)	562
ELQLRL	Length of low priority request/response units leaving the transmission priority queue.	PD (11,0)	568
ELQTRR	Cumulative wait time in low transmission priority queue.	PD (11,0)	574
ELNRUD	Number of low priority request/response units delivered to the adjacent system.	PD (11,0)	580
ELLRUD	Length of low priority request/response units delivered to the adjacent system.	PD (11,0)	586
ELTRUD	Cumulative service time to deliver a low priority request/response unit to the adjacent system.	PD (11,0)	592
ELNRUR	Number of low priority request/response units received from the adjacent system.	PD (11,0)	598
ELLRUR	Length of low priority request/response units received from the adjacent system.	PD (11,0)	604
	The following fields refer to intermediate sessions.		
INNSS	Number of network priority sessions started	PD (11,0)	610
INNSE	Number of network priority sessions ended	PD (11,0)	616
INNBB	Number of request units with begin bracket sent and received for all network priority sessions	PD (11,0)	622
INNEB	Number of request units with end bracket sent and received for all network priority sessions	PD (11,0)	628

Figure A-29 (Page 7 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
INSPWT	The cumulative wait time for all network priority sessions (in milliseconds) caused by session-level send messages. This wait time measures the amount of time application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system	PD (11,0)	634
INSPNW	Number of waits occurring for all network priority sessions for session-level send pacing. That is, the number of times application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system	PD (11,0)	640
INSPPW	Number of potential waits occurring for all network priority sessions for session-level send pacing. This is the worst case that would occur if the sending of application data was delayed waiting for every pacing response sent by the adjacent system.	PD (11,0)	646
INSPWS	The cumulative window size for all network priority sessions for session-level send pacing. Each time a pacing response is received from the adjacent system on a network priority session, this count is increased by window size specified by the pacing response.	PD (11,0)	652
INIPWT	The cumulative wait time for all network priority sessions (in milliseconds) for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	658
INIPNW	Number of waits occurring for all network priority sessions for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	664
INQNRE	Number of network priority request/response units entering the transmission priority queue.	PD (11,0)	670
INQLRE	Length of network priority request/response units entering the transmission priority queue.	PD (11,0)	676
INQNRL	Number of network priority request/response units leaving the transmission priority queue.	PD (11,0)	682
INQLRL	Length of network priority request/response units leaving the transmission priority queue.	PD (11,0)	688
INQTRR	Cumulative wait time in network transmission priority queue.	PD (11,0)	694
INNRUD	Number of network priority request/response units delivered to the adjacent system.	PD (11,0)	700
INLRUD	Length of network priority request/response units delivered to the adjacent system.	PD (11,0)	706
INTRUD	Cumulative service time to deliver a network priority request/response unit to the adjacent system.	PD (11,0)	712
INNRUR	Number of network priority request/response units received from the adjacent system.	PD (11,0)	718
INLRUR	Length of network priority request/response units received from the adjacent system.	PD (11,0)	724
IHNSS	Number of high priority sessions started.	PD (11,0)	730
IHNSE	Number of high priority sessions ended.	PD (11,0)	736

Figure A-29 (Page 8 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
IHNBB	Number of request units with begin bracket sent and received for all high priority sessions.	PD (11,0)	742
IHNEB	Number of request units with end bracket sent and received for all high priority sessions.	PD (11,0)	748
IHSPWT	The cumulative wait time for all high priority sessions (in milliseconds) caused by session-level send messages. This wait time measures the amount of time application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system.	PD (11,0)	754
IHSPNW	Number of waits occurring for all high priority sessions for session-level send pacing. That is, the number of times application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system.	PD (11,0)	760
IHSPPW	Number of potential waits occurring for all high priority sessions for session-level send pacing. This is the worst case that would occur if the sending of application data was delayed waiting for every pacing response sent by the adjacent system.	PD (11,0)	766
IHSPWS	The cumulative window size for all high priority sessions for session-level send pacing. Each time a pacing response is received from the adjacent system on a network priority session, this count is increased by window size specified by the pacing response.	PD (11,0)	772
IHIPWT	The cumulative wait time for all high priority sessions (in milliseconds) for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	778
IHIPNW	Number of waits occurring for all high priority sessions for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	784
IHQNRE	Number of high priority request/response units entering the transmission priority queue.	PD (11,0)	790
IHQLRE	Length of high priority request/response units entering the transmission priority queue.	PD (11,0)	796
IHQNRL	Number of high priority request/response units leaving the transmission priority queue.	PD (11,0)	802
IHQLRL	Length of high priority request/response units leaving the transmission priority queue.	PD (11,0)	808
IHQTRR	Cumulative wait time in high transmission priority queue.	PD (11,0)	814
IHNRUD	Number of high priority request/response units delivered to the adjacent system.	PD (11,0)	820
IHLRUD	Length of high priority request/response units delivered to the adjacent system.	PD (11,0)	826
IHTRUD	Cumulative service time to deliver a high priority request/response unit to the adjacent system.	PD (11,0)	832
IHNRUR	Number of high priority request/response units received from the adjacent system.	PD (11,0)	838
IHLRUR	Length of high priority request/response units received from the adjacent system.	PD (11,0)	844

Figure A-29 (Page 9 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
IMNSS	Number of medium priority sessions started.	PD (11,0)	850
IMNSE	Number of medium priority sessions ended.	PD (11,0)	856
IMNBB	Number of request units with begin bracket sent and received for all medium priority sessions.	PD (11,0)	862
IMNEB	Number of request units with end bracket sent and received for all medium priority sessions.	PD (11,0)	868
IMSPWT	The cumulative wait time for all medium priority sessions (in milliseconds) caused by session-level send messages. This wait time measures the amount of time application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system.	PD (11,0)	874
IMSPNW	Number of waits occurring for all medium priority sessions for session-level send pacing. That is, the number of times application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system.	PD (11,0)	880
IMSPPW	Number of potential waits occurring for all medium priority sessions for session-level send pacing. This is the worst case that would occur if the sending of application data was delayed waiting for every pacing response sent by the adjacent system.	PD (11,0)	886
IMSPWS	The cumulative window size for all medium priority sessions for session-level send pacing. Each time a pacing response is received from the adjacent system on a network priority session, this count is increased by window size specified by the pacing response.	PD (11,0)	892
IMIPWT	The cumulative wait time for all medium priority sessions (in milliseconds) for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	898
IMIPNW	Number of waits occurring for all medium priority sessions for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	904
IMQNRE	Number of medium priority request/response units entering the transmission priority queue.	PD (11,0)	910
IMQLRE	Length of medium priority request/response units entering the transmission priority queue.	PD (11,0)	916
IMQNRL	Number of medium priority request/response units leaving the transmission priority queue.	PD (11,0)	922
IMQLRL	Length of medium priority request/response units leaving the transmission priority queue.	PD (11,0)	928
IMQTRR	Cumulative wait time in medium transmission priority queue.	PD (11,0)	934
IMNRUD	Number of medium priority request/response units delivered to the adjacent system.	PD (11,0)	940
IMLRUD	Length of medium priority request/response units delivered to the adjacent system.	PD (11,0)	946
IMTRUD	Cumulative service time to deliver a medium priority request/response unit to the adjacent system.	PD (11,0)	952

Figure A-29 (Page 10 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
IMNRUR	Number of medium priority request/response units received from the adjacent system.	PD (11,0)	958
IMLRUR	Length of medium priority request/response units received from the adjacent system.	PD (11,0)	964
ILNSS	Number of low priority sessions started.	PD (11,0)	970
ILNSE	Number of low priority sessions ended.	PD (11,0)	976
ILNBB	Number of request units with begin bracket sent and received for all low priority sessions.	PD (11,0)	982
ILNEB	Number of request units with end bracket sent and received for all low priority sessions.	PD (11,0)	988
ILSPWT	The cumulative wait time for all low priority sessions (in milliseconds) caused by session-level send messages. This wait time measures the amount of time application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system.	PD (11,0)	994
ILSPNW	Number of waits occurring for all low priority sessions for session-level send pacing. That is, the number of times application data was blocked (could not be sent) waiting for a pacing response to be received from the adjacent system.	PD (11,0)	1000
ILSPPW	Number of potential waits occurring for all low priority sessions for session-level send pacing. This is the worst case that would occur if the sending of application data was delayed waiting for every pacing response sent by the adjacent system.	PD (11,0)	1006
ILSPWS	The cumulative window size for all low priority sessions for session-level send pacing. Each time a pacing response is received from the adjacent system on a network priority session, this count is increased by window size specified by the pacing response.	PD (11,0)	1012
ILIPWT	The cumulative wait time for all low priority sessions (in milliseconds) for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	1018
ILIPNW	Number of waits occurring for all low priority sessions for internal session-level pacing. That is, the number of times application data was blocked (could not be sent) waiting for data to be delivered to the adjacent system.	PD (11,0)	1024
ILQNRE	Number of low priority request/response units entering the transmission priority queue.	PD (11,0)	1030
ILQLRE	Length of low priority request/response units entering the transmission priority queue.	PD (11,0)	1036
ILQNRL	Number of low priority request/response units leaving the transmission priority queue.	PD (11,0)	1042
ILQLRL	Length of low priority request/response units leaving the transmission priority queue.	PD (11,0)	1048
ILQTRR	Cumulative wait time in low transmission priority queue.	PD (11,0)	1054
ILNRUD	Number of low priority request/response units delivered to the adjacent system. PD (11,0)	1054	

Figure A-29 (Page 11 of 11). SNA Data (One for Each Active T2 Task)

Field Name	Description	Attributes	Buffer Position
ILLRUD	Length of low priority request/response units delivered to the adjacent system.	PD (11,0)	1066
ILTRUD	Cumulative service time to deliver a low priority request/response unit to the adjacent system.	PD (11,0)	1072
ILNRUR	Number of low priority request/response units received from the adjacent system.	PD (11,0)	1078
ILLRUR	Length of low priority request/response units received from the adjacent system.	PD (11,0)	1084

File Name: QAPMSNADS

SNADS Data: Figure A-30 lists the fields in the SNA distributions services (SNADS) file.

Figure A-30 (Page 1 of 2). SNADS Data (One Entry Per SNADS Job)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
SNJNAM	SNADS job name.	C(10)	20
SNJUSR	SNADS job user.	C(10)	30
SNJNBR	SNADS job number.	C(6)	40
SNFTYP	This is a SNADS function type indicating which SNADS function this job is running (see note 1).	PD(3,0)	46
SNNTR	Transaction count.	PD(11,0)	48
SNTRT	Transaction time: The time from a distribution being put on the queue to the time processing that distribution within this job is completed.	PD(11,0)	54
SNRUT	Resource usage time: The total time that distributions are processed, not including time that they are waiting on the queue.	PD(11,0)	60
SNATN	Active transitions: The number of transitions between waiting for conditions to be satisfied (a distribution to process) and starting to process a distribution.	PD(11,0)	66
SNERR	Error count: Number of transactions that ended in error.	PD(11,0)	72
SNNRC	Number of recipients: The number of recipients identified in the distribution.	PD(11,0)	78
SNFSO	File server object (FSO) count: The number of transactions that required a data object or document to be processed.	PD(11,0)	84
SNFSOB	FSO byte count: The size of the FSOs (data objects and documents) processed by transactions.	PD (11,0)	90

Figure A-30 (Page 2 of 2). SNADS Data (One Entry Per SNADS Job)

Field Name	Description	Attributes	Buffer Position
SNFOC	Fan-out count: The accumulated value of the number of distribution queues that received a copy of a distribution during routing. For a single distribution processed by the router, this value is the number of sender transactions (paths) the distribution will take leaving the system. This is the number of distribution copies that leave the system. (This field is only supported by the router job.)	PD (11,0)	96
SNLDC	Set to '1' when a local delivery queue received a copy of the distribution during routing. This indicates that the local system was a destination for the distribution. (This field is only supported by the router job.)	PD(11,0)	102

Note:

The SNFTYP field is used to determine the type of activity that this SNADS job conducts.

SNFTYP = 1 for SNADS router
SNFTYP = 2 for SNADS receiver
SNFTYP = 3 for SNADS sender
SNFTYP = 8 for SNADS DLS Gate (Document Library Services)
SNFTYP = 9 for SNADS RPDS Gate (VM/MVS bridge, SMTP, X.400)

File Name: QAPMAPPN

T2 task data: Figure A-31 lists the fields in the Advanced Peer-to-Peer Network (APPN) data file.

Figure A-31 (Page 1 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
INTNUM	Interval number: The nth sample interval since the start of the performance monitor job.	PD (5,0)	1
DTETIM	Interval date (yy/mm/dd) and time (hh:mm:ss): The date and time of the sample interval.	C (12)	4
INTSEC	Elapsed interval seconds: The number of seconds since the last sample interval.	PD (7,0)	16
ANTGU	Total number of transmission group (TG) updates processed	PD(11,0)	20
ATTGU	Cumulative time (in milliseconds) to process the TG updates	PD(11,0)	26
ANTGUM	Number of TG updates that require one or more resources to be added to the topology database update (TDU) buffer	PD(11,0)	32
ANRATG	Number of resources added to TDU buffers due to TG update processing	PD(11,0)	38
ANTSTG	Number of TDUs sent as a result of initially creating a TDU buffer on behalf of TG updates	PD(11,0)	44
ANNTTG	Number of network nodes that had TDUs sent to them due to TDUs being created for TG update processing	PD(11,0)	50
ANNCTC	Total number of node congestion transition changes processed	PD(11,0)	56
ATNCTC	Cumulative elapsed time for processing congestion transition changes	PD(11,0)	62
ATRSNC	Number of times that topology routing services (TRS) entered into non-congested state	PD (11,0)	68
ATRSC	Number of times that TRS entered into congested state	PD (11,0)	74

Figure A-31 (Page 2 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
ATNCS	Cumulative elapsed time (in milliseconds) that the system was in non-congested state	PD(11,0)	80
ATCS	Cumulative elapsed time (in milliseconds) that the system was in congested state	PD (11,0)	86
ATSCP	Number of TDUs sent as a result of initially creating a TDU buffer on behalf of node congestion processing	PD (11,0)	92
ANTSCP	Number of network nodes that had TDUs sent to them due to TDUs being created for node congestion processing	PD (11,0)	98
ANTDUP	Total number of received TDUs processed by this node	PD (11,0)	104
ATTDUP	Cumulative elapsed time for processing the received TDUs	PD (11,0)	110
ANNRTD	Number of new resources received in TDUs that cause resources to be added to the TDU buffer	PD (11,0)	116
ANORTN	Number of old resources received in TDUs that do not require a resource to be added to the TDU buffer	PD (11,0)	122
ANORTA	Number of old resources received in TDUs that do require resources to be added to the TDU buffer	PD (11,0)	128
ANTSRT	Number of TDUs sent as a result of initially creating a TDU buffer on behalf of processing a received TDU	PD (11,0)	134
ANNTST	Number of network nodes that had TDUs sent to them due to TDUs being created for processing received TDUs	PD (11,0)	140
ACNTID	Network ID of the node that received the most TDUs within the interval	C (8)	146
ACCPNM	Control point (CP) name of the node that received the most TDUs within the interval	C (8)	154
ANTRFN	Number of TDUs received this interval by the node that received the most TDUs in the interval	PD (11,0)	162
ANITEP	Total number of initial topology exchanges processed by this node	PD (11,0)	168
ATPIE	Cumulative elapsed time for processing the initial exchange	PD (11,0)	174
ANTECT	Number of times the initial topology exchange caused the complete network node topology to be sent	PD (11,0)	180
ANTDE	Total number of entries in the entire topology database (this value is not a delta)	PD (11,0)	186
ANTERS	Number of resources (nodes and TGs) added to the TDU buffer due to initial topology exchange	PD (11,0)	192
ANTETS	Number of TDUs sent as a result of initial topology exchange	PD (11,0)	198
ANGCP	Number of times that obsolete topology entries were removed	PD (11,0)	204
ATGCP	Cumulative elapsed time for removing the obsolete topology entries	PD (11,0)	210
ANTEDG	Number of topology entries that were deleted	PD (11,0)	216
ANTGC	Number of TDUs that were sent when obsolete topology entries were deleted	PD (11,0)	222
ANNTGC	Number of network nodes that had TDUs sent to them when obsolete topology entries were removed	PD (11,0)	228
ANRRP	Total number of registration requests processed	PD (11,0)	234
ANNLRR	Total number of locations processed via the registration requests	PD (11,0)	240

Figure A-31 (Page 3 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
ATPRR	Cumulative elapsed time to process registration requests	PD (11,0)	246
ANDRP	Total number of deletion requests processed	PD (11,0)	252
ANLDDR	Total number of locations deleted via the deletion requests	PD (11,0)	258
ATPDR	Cumulative elapsed time to process the deletion requests	PD (11,0)	264
ANCNAP	Total number of requests to change network attributes processed	PD (11,0)	270
ATCNA	Cumulative elapsed time to process the requests to change network attributes	PD (11,0)	276
ANDDRC	Number of times the directory database was deleted and re-created due to processing the requests to change network attributes	PD (11,0)	282
ANLRSC	Number of location registration requests sent due to processing the requests to change network attributes	PD (11,0)	288
ANLDSC	Number of location deletion requests sent due to processing the requests to change network attributes	PD (11,0)	294
ANTDRC	Number of times the topology database was deleted and re-created due to processing the requests to change network attributes	PD (11,0)	300
ANCART	Number of time the requests to change network attributes caused a node entry resource to be added to the TDU buffer	PD (11,0)	306
ANTSTC	Number of TDUs sent as a result of initially creating a TDU buffer on behalf of requests to change network attributes	PD (11,0)	312
ANNTSC	Number of network nodes that had TDUs sent to them due to TDUs being created for processing requests to change network attributes	PD (11,0)	318
ANDAII	Number of times APPN information was displayed (DSPAPPNINF command)	PD (11,0)	324
ANLLUP	Total number of local location list updates processed	PD (11,0)	330
ATLLUP	Cumulative elapsed time to process the local location list updates	PD (11,0)	336
ANLRSL	Number of location registration requests sent due to local location list updates	PD (11,0)	342
ANLDLL	Number of location deletion requests sent due to local location list updates	PD (11,0)	348
ANRLUP	Total number of remote location list updates processed	PD (11,0)	354
ATRLUP	Cumulative elapsed time to process the remote location list updates	PD (11,0)	360
ANMDUP	Total number of mode description updates processed by APPN	PD (11,0)	366
ATMDUP	Cumulative elapsed time to process the mode description updates	PD (11,0)	372
ANCSUP	Total number of class-of-service updates processed by APPN	PD (11,0)	378
ATCSUT	Cumulative elapsed time to process the class-of-service (COS) update by TRS	PD (11,0)	384
ATCSUC	Cumulative elapsed time to process the COS update by the CPMGR task	PD (11,0)	390
ANCSSA	Number of contention CP-CP session setups attempted	PD (11,0)	396
ANCSSS	Number of contention CP-CP session setups successful	PD (11,0)	402
ANRRS	Total number of registration requests sent	PD (11,0)	408
ANLRRR	Total number of locations registered via the registration requests	PD (11,0)	414

Figure A-31 (Page 4 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
ATSRR	Cumulative elapsed time to send the registration requests	PD (11,0)	420
ANSTC	Number of single-hop route requests made to TRS for contention CP session setup	PD (11,0)	426
ANSTCS	Number of single-hop route requests made to topology routing services (TRS) for contention CP session setup that were successful	PD (11,0)	432
ATSTCS	Cumulative elapsed time for processing single-hop route requests on behalf of contention CP session setups	PD (11,0)	438
ANARMC	Number of activate-route requests made to MSCP for contention CP session setups	PD (11,0)	444
ANSARM	Number of successful activate-route requests processed by MSCP for contention CP session setups	PD (11,0)	450
ATARMC	Cumulative elapsed time for activate-route requests on behalf of contention CP session setups	PD (11,0)	456
ANTDSC	Number of requests made to the T2 SIOM to perform device selection on behalf of contention CP session setups	PD (11,0)	462
ATTDSC	Cumulative elapsed time for device selection processing to complete on behalf of contention CP session setups	PD (11,0)	468
ANDSS	Number of device selection requests that were successful on behalf of contention CP session setups	PD (11,0)	474
ATCCSA	Cumulative elapsed time for processing contention CP session activation requests	PD (11,0)	480
ANLSAP	Number of contention CP session activations processed	PD (11,0)	486
ANCST	Number of contention CP-CP session ends	PD (11,0)	492
ATCST	Cumulative elapsed time for processing contention CP-CP session ends	PD (11,0)	498
ANLST	Number of contention CP-CP session ends	PD (11,0)	504
ATLST	Cumulative elapsed time for processing contention CP-CP session ends	PD (11,0)	510
ANCWSA	Number of contention CP-CP sessions currently active (this is not a delta)	PD (11,0)	516
ANCLSA	Number of contention CP-CP sessions currently active (this is not a delta)	PD (11,0)	522
ANCDRR	Number of data-received requests processed (CP capabilities)	PD (11,0)	528
ANCBDR	Number of bytes of data received (CP capabilities)	PD (11,0)	534
ATCDRR	Cumulative elapsed time for processing the data-received requests (CP capabilities)	PD (11,0)	540
ANCSDR	Number of send-data requests processed (CP capabilities)	PD (11,0)	546
ANCBDS	Number of bytes of data sent through the send-data requests (CP capabilities)	PD (11,0)	552
ATCSDR	Cumulative elapsed time for processing the send-data requests (CP capabilities)	PD (11,0)	588
ANTDRR	Number of data-received requests processed (topology database update)	PD (11,0)	564

Figure A-31 (Page 5 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
ANTBDR	Number of bytes of data received (topology database update)	PD (11,0)	570
ATTDRR	Cumulative elapsed time for processing the data-received requests (topology database update)	PD (11,0)	576
ANTSDR	Number of send-data requests processed (topology database update)	PD (11,0)	582
ANTBDS	Number of bytes of data sent through the send-data requests (topology database update)	PD (11,0)	588
ATTSDR	Cumulative elapsed time for processing the send-data requests (topology database update)	PD (11,0)	594
ANDDRR	Number of data-received requests processed (directory search)	PD (11,0)	600
ANDBDR	Number of bytes of data received (directory search)	PD (11,0)	606
ATDDRR	Cumulative elapsed time for processing the data-received requests (directory search)	PD (11,0)	612
ANDSDR	Number of send-data requests processed (directory search)	PD (11,0)	618
ANDBDS	Number of bytes of data sent by the send-data requests (directory search)	PD (11,0)	624
ATDSDR	Cumulative elapsed time for processing the send-data requests (directory search)	PD (11,0)	630
ANRDRR	Number of data-received requests processed (registration/deletion)	PD (11,0)	636
ANRBDR	Number of bytes of data received (registration/deletion)	PD (11,0)	642
ATRDRR	Cumulative elapsed time for processing the data-received requests (registration/deletion)	PD (11,0)	648
ANRSDR	Number of send-data requests processed (registration/deletion)	PD (11,0)	654
ANRBDS	Number of bytes of data sent through the send-data requests (registration/deletion)	PD (11,0)	660
ATRSDR	Cumulative elapsed time for processing the send-data requests (registration/deletion)	PD (11,0)	666
	Local system initiated sessions		
ANWAP1	Total number of work activities of this type processed	PD (11,0)	672
ATWAP1	Cumulative elapsed time to complete work activities of this type	PD (11,0)	678
ATWAS1	Total number of work activities of this type that yielded a successful result	PD (11,0)	684
ASSSA1	Number of session setup attempts satisfied through an existing APPN session	PD (11,0)	690
AASNA1	Number of APPC session requests satisfied by using non-APPN device descriptions	PD (11,0)	696
ASPAC1	Number of session setup requests that require APPN control point services for directory, route selection, and device selection processing	PD (11,0)	702
ASPSP1	Number of session setup requests that get pended due to another session setup being in progress for the same local location, remote location, and mode	PD (11,0)	708
ASLNS1	Number of searches that the local end node satisfied locally (that is, without sending a search to its network node (NN) server)	PD (11,0)	714
AS1HS1	Number of one-hop search requests sent by the end node (EN)	PD (11,0)	720

Figure A-31 (Page 6 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
A1HSS1	Number of searches satisfied by the end node by sending one-hop search requests	PD (11,0)	726
ASSBN1	Number of searches satisfied by sending a bind directly to an attached network node server (because the end node has no CP-CP session to a server)	PD (11,0)	732
ASFNS1	Number of searches that failed because of no network services being available for the local end node	PD (11,0)	738
ATILP1	Cumulative elapsed time required for the locate phase initiated by the end node to complete	PD (11,0)	744
ANSSL1	Number of searches satisfied locally (using the topology database or the directory services (DS) database and finding an entry for an end node that does not support CP sessions)	PD (11,0)	750
ANIHS1	Number of one-hop search requests sent by the network node	PD (11,0)	756
ANSS11	Number of searches satisfied by the network node by sending one-hop search requests	PD (11,0)	762
ANDSS1	Number of directed searches sent	PD (11,0)	768
ASSDS1	Number of searches that were satisfied by sending directed searches	PD (11,0)	774
ATDSR1	Cumulative elapsed time for directed search responses to be received	PD (11,0)	780
ANDBE1	Number of domain broadcasts that have been run	PD (11,0)	786
ANNDB1	Number of nodes that these domain broadcasts have been sent to	PD (11,0)	792
ATRDB1	Cumulative elapsed time for the first positive response to be returned on domain broadcasts	PD (11,0)	798
ATLRD1	Cumulative elapsed time for the last response to be returned on domain broadcasts	PD (11,0)	804
ASSDB1	Number of searches that were satisfied by sending a domain broadcast	PD (11,0)	810
ANBSE1	Number of broadcast searches that have been run	PD (11,0)	816
ANNBS1	Number of adjacent nodes that these broadcast searches have been sent to	PD (11,0)	822
ATRBS1	Cumulative elapsed time for the first positive response to be returned on broadcast searches	PD (11,0)	828
ATLRB1	Cumulative elapsed time for the last response to be returned on broadcast searches	PD (11,0)	834
ANSBS1	Number of searches that were satisfied by sending a broadcast search	PD (11,0)	840
ATSPR1	Cumulative elapsed time from the start of search processing on the local node until a positive response has been returned to the local user	PD (11,0)	846
ATSPC1	Cumulative elapsed time from the start of search processing until the local directory services task has completed all processing for the request. This measurement takes into account the time required to process domain broadcast or broadcast search responses even though a positive response has already been sent back to the local user	PD (11,0)	852
AN1HT1	Number of single-hop route requests made to topology routing services (TRS)	PD (11,0)	858
AS1HT1	Number of single-hop route requests made to TRS that were successful	PD (11,0)	864

Figure A-31 (Page 7 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
AT1HC1	Cumulative elapsed time for processing single-hop route requests	PD (11,0)	870
ANRRT1	Number of request-route requests made to TRS	PD (11,0)	876
ASRRT1	Number of request-route requests made to TRS that were successful	PD (11,0)	882
ATRRT1	Cumulative elapsed time for processing request-route requests	PD (11,0)	888
AARRM1	Number of activate-route requests made to machine services control point (MSCP)	PD (11,0)	894
AARCV1	Number of activate-route requests that require a controller description to be automatically created and/or varied on by the system	PD (11,0)	900
ATRCV1	Cumulative elapsed time for automatic creation and/or vary on of the controller to be processed	PD (11,0)	906
ASARR1	Number of successful activate-route requests processed by MSCP	PD (11,0)	912
ATARP1	Cumulative elapsed time for processing activate-route requests by MSCP	PD (11,0)	918
ARDS1	Number of requests made to the T2 SIOM to perform device selection	PD (11,0)	924
ATDS1	Cumulative elapsed time for device selection processing to complete	PD (11,0)	930
ADSS1	Number of device selection requests that were successful	PD (11,0)	936
	Receiver of search requests as an end node		
ANWAP2	Total number of work activities of this type processed	PD (11,0)	942
ATWAP2	Cumulative elapsed time to complete work activities of this type	PD (11,0)	948
ATWAS2	Total number of work activities of this type that yielded a successful result	PD (11,0)	954
	Network node performing search requests on behalf of an end node		
ANWAP3	Total number of work activities of this type processed	PD (11,0)	960
ATWAP3	Cumulative elapsed time to complete work activities of this type	PD (11,0)	966
ATWAS3	Total number of work activities of this type that yielded a successful result	PD (11,0)	972
ANSSL3	Number of searches satisfied locally (by referring to the topology database or by using the directory services database and finding an entry for an end node that does not support control point sessions)	PD (11,0)	978
ANIHS3	Number of one-hop search requests sent by the network node	PD (11,0)	984
ANSS13	Number of searches satisfied by the network node by sending one-hop search requests	PD (11,0)	990
ANDSS3	Number of directed searches sent	PD (11,0)	996
ASSDS3	Number of searches that were satisfied by sending directed searches	PD (11,0)	1002
ATDSR3	Cumulative elapsed time for directed search responses to be received	PD (11,0)	1008
ANDBE3	Number of domain broadcasts that have been run	PD (11,0)	1014
ANNDB3	Number of nodes that these domain broadcasts have been sent to	PD (11,0)	1020
ATRDB3	Cumulative elapsed time for the first positive response to be returned on domain broadcasts	PD (11,0)	1026
ATLRD3	Cumulative elapsed time for the last response to be returned on domain broadcasts	PD (11,0)	1032

Figure A-31 (Page 8 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
ASSDB3	Number of searches that were satisfied by sending a domain broadcast	PD (11,0)	1038
ANBSE3	Number of broadcast searches that have been run	PD (11,0)	1044
ANNBS3	Number of adjacent nodes that these broadcast searches have been sent to	PD (11,0)	1050
ATRBS3	Cumulative elapsed time for the first positive response to be returned on broadcast searches	PD (11,0)	1056
ATLRB3	Cumulative elapsed time for the last response to be returned on broadcast searches	PD (11,0)	1062
ANSBS3	Number of searches that were satisfied by sending a broadcast search	PD (11,0)	1068
ATSPR3	Cumulative elapsed time from the start of search processing on the local node until a response has been returned to the local user or remote system that initiated the search process on the local system	PD (11,0)	1074
ATSPC3	Cumulative elapsed time from the start of search processing until the local directory services task has completed all processing for the request. This measurement takes into account the time required to process domain broadcast or broadcast search responses even though a positive response has already been sent back to the local user or remote system that initiated a search	PD (11,0)	1080
ANRRT3	Number of request-route requests made to TRS	PD (11,0)	183
ASRRT3	Number of request-route requests made to TRS that were successful	PD (11,0)	1086
ATRRT3	Cumulative elapsed time for processing request-route requests	PD (11,0)	1098
	Intermediate node on a directed search request		
ANWAP4	Total number of work activities of this type processed	PD (11,0)	1104
ATWAP4	Cumulative elapsed time to complete work activities of this type	PD (11,0)	1110
ATWAS4	Total number of work activities of this type that yielded a successful result	PD (11,0)	1116
	Network node that is the destination node of a directed search request		
ANWAP5	Total number of work activities of this type processed	PD (11,0)	1122
ATWAP5	Cumulative elapsed time to complete work activities of this type	PD (11,0)	1128
ATWAS5	Total number of work activities of this type that yielded a successful result	PD (11,0)	1134
ANSSL5	Number of searches satisfied locally (by referring to the topology database or by using the directory services database and finding an entry for an end node that does not support control point sessions)	PD (11,0)	1140
ANIHS5	Number of one-hop search requests sent by the network node	PD (11,0)	1146
ANSS15	Number of searches satisfied by the network node by sending one-hop search requests	PD (11,0)	1152
ANDBE5	Number of domain broadcasts that have been run	PD (11,0)	1158
ANNDB5	Number of nodes that these domain broadcasts have been sent to	PD (11,0)	1164
ATRDB5	Cumulative elapsed time for the first positive response to be returned on domain broadcasts	PD (11,0)	1170
ATLRD5	Cumulative elapsed time for the last response to be returned on domain broadcasts	PD (11,0)	1176

Figure A-31 (Page 9 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
ASSDB5	Number of searches that were satisfied by sending a domain broadcast	PD (11,0)	1182
	Network node processing a received-broadcast-search request		
ANWAP6	Total number of work activities of this type processed	PD (11,0)	1188
ATWAP6	Cumulative elapsed time to complete work activities of this type	PD (11,0)	1194
ATWAS6	Total number of work activities of this type that yielded a successful result	PD (11,0)	1200
ANSSL6	Number of searches satisfied locally (by referring to the topology database or by using the directory services database and finding an entry for an end node that does not support control point sessions)	PD (11,0)	1206
ANIHS6	Number of one-hop search requests sent by the network node	PD (11,0)	1212
ANSS16	Number of searches satisfied by the network node by sending one-hop search requests	PD (11,0)	1218
ANDBE6	Number of domain broadcasts that have been run	PD (11,0)	1224
ANNDB6	Number of nodes that these domain broadcasts have been sent to	PD (11,0)	1230
ATRDB6	Cumulative elapsed time for the first positive response to be returned on domain broadcasts	PD (11,0)	1236
ATLRD6	Cumulative elapsed time for the last response to be returned on domain broadcasts	PD (11,0)	1242
ASSDB6	Number of searches that were satisfied by sending a domain broadcast	PD (11,0)	1248
	Network node processing a received-search request from a node in a non-AS/400 network		
ANWAP7	Total number of work activities of this type processed	PD (11,0)	1254
ATWAP7	Cumulative elapsed time to complete work activities of this type	PD (11,0)	1260
ATWAS7	Total number of work activities of this type that yielded a successful result	PD (11,0)	1266
ANSSL7	Number of searches satisfied locally (by referring to the topology database or by using the directory services database and finding an entry for an end node that does not support control point sessions)	PD (11,0)	1272
ANIHS7	Number of one-hop search requests sent by the network node	PD (11,0)	1278
ANSS17	Number of searches satisfied by the network node by sending one-hop search requests	PD (11,0)	1284
ANDSS7	Number of directed searches sent	PD (11,0)	1290
ASSDS7	Number of searches that were satisfied by sending directed searches	PD (11,0)	1296
ATDSR7	Cumulative elapsed time for directed search responses to be used	PD (11,0)	1302
ANDBE7	Number of domain broadcasts that have been run	PD (11,0)	1308
ANNDB7	Number of nodes that these domain broadcasts have been sent to	PD (11,0)	1314
ATRDB7	Cumulative elapsed time for the first positive response to be returned on domain broadcasts	PD (11,0)	1320
ATLRD7	Cumulative elapsed time for the last response to be returned on domain broadcasts	PD (11,0)	1326
ASSDB7	Number of searches that were satisfied by sending a domain broadcast	PD (11,0)	1332
ANBSE7	Number of broadcast searches that have been run	PD (11,0)	1338

Figure A-31 (Page 10 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
ANNBS7	Number of adjacent nodes that these broadcast searches have been sent to	PD (11,0)	1344
ATRBS7	Cumulative elapsed time for the first positive response to be returned on broadcast searches	PD (11,0)	1350
ATLRB7	Cumulative elapsed time for the last response to be returned on broadcast searches	PD (11,0)	1356
ANSBS7	Number of searches that were satisfied by sending a broadcast search	PD (11,0)	1362
ATSPR7	Cumulative elapsed time from the start of search processing on the local node until a response has been returned to the remote system that initiated the search process on the local system	PD (11,0)	1368
ATSPC7	Cumulative elapsed time from the start of search processing until the local directory services task has completed all processing for the request. This measurement takes into account the time required to process domain broadcast or broadcast search responses even though a positive response has already been sent back to the remote system that initiated a search	PD (11,0)	1374
ANRRT7	Number of request-route requests made to topology routing services (TRS)	PD (11,0)	1380
ASRRT7	Number of request-route requests made to topology routing services (TRS) that were successful	PD (11,0)	1386
ATRRT7	Cumulative elapsed time for processing request-route requests	PD (11,0)	1392
	Network node processing a received-bind request from a node in the AS/400 network without routing information		
ANWAP8	Total number of work activities of this type processed	PD (11,0)	1398
ATWAP8	Cumulative elapsed time to complete work activities of this type	PD (11,0)	1404
ATWAS8	Total number of work activities of this type that yielded a successful result	PD (11,0)	1410
ASPSP8	Number of session setup requests that are placed in pending status due to another session setup being in progress for the same local location, remote location, and mode	PD (11,0)	1416
ANSSL8	Number of searches satisfied locally (by referring to the topology database or by using the directory services database and finding an entry for an end node that does not support control point sessions)	PD (11,0)	1422
ANIHS8	Number of one-hop search requests sent by the network node	PD (11,0)	1428
ANSS18	Number of searches satisfied by the network node by sending one-hop search requests	PD (11,0)	1434
ANDSS8	Number of directed searches sent	PD (11,0)	1440
ASSDS8	Number of searches that were satisfied by sending directed searches	PD (11,0)	1446
ATDSR8	Cumulative elapsed time for directed search responses to be used	PD (11,0)	1458
ANDBE8	Number of domain broadcasts that have been run	PD (11,0)	1464
ANNDB8	Number of nodes that these domain broadcasts have been sent to	PD (11,0)	1464
ATRDB8	Cumulative elapsed time for the first positive response to be returned on domain broadcasts	PD (11,0)	1470

Figure A-31 (Page 11 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
ATLRD8	Cumulative elapsed time for the last response to be returned on domain broadcasts	PD (11,0)	1476
ASSDB8	Number of searches that were satisfied by sending a domain broadcast	PD (11,0)	1482
ANBSE8	Number of broadcast searches that have been run	PD (11,0)	1488
ANNBS8	Number of adjacent nodes that these broadcast searches have been sent to	PD (11,0)	1494
ATRBS8	Cumulative elapsed time for the first positive response to be returned on broadcast searches	PD (11,0)	1500
ATLRB8	Cumulative elapsed time for the last response to be returned on broadcast searches	PD (11,0)	1508
ANSBS8	Number of searches that were satisfied by sending a broadcast search	PD (11,0)	1512
ATSPR8	Cumulative elapsed time from the start of search processing on the local node until a response has been returned to the local system to allow the bind processing to continue	PD (11,0)	1518
ATSPC8	Cumulative elapsed time from the start of search processing until the local directory services task has completed all processing for the request. This measurement takes into account the time required to process domain broadcast or broadcast search responses even though a positive response has already been sent back to the local system to allow the bind processing to continue	PD (11,0)	1524
ANRRT8	Number of request-route requests made to topology routing services (TRS)	PD (11,0)	1530
ASRRT8	Number of request-route requests made to TRS that were successful	PD (11,0)	1536
ATRRT8	Cumulative elapsed time for processing request-route requests	PD (11,0)	1542
AARRM8	Number of activate-route requests made to machine services control point (MSCP)	PD (11,0)	1548
AARCV8	Number of activate-route requests that require a controller description to be automatically created and/or varied on by the system	PD (11,0)	1554
ATRCV8	Cumulative elapsed time for automatic creation and/or vary on of the controller to be processed	PD (11,0)	1560
ASARR8	Number of successful activate-route requests processed by MSCP	PD (11,0)	1566
ATARP8	Cumulative elapsed time for processing activate-route requests by MSCP	PD (11,0)	1572
	Network node processing a received-bind request from a node in a non-AS/400 network without routing information		
ANWAP9	Total number of work activities of this type processed	PD (11,0)	1578
ATWAP9	Cumulative elapsed time to complete work activities of this type	PD (11,0)	1584
ATWAS9	Total number of work activities of this type that yielded a successful result	PD (11,0)	1590
ASPSP9	Number of session setup requests that are placed in pending status due to another session setup being in progress for the same local location, remote location, and mode	PD (11,0)	1596

Figure A-31 (Page 12 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
ANSSL9	Number of searches satisfied locally (by referring to the topology database or by using the directory services database and finding an entry for an end node that does not support control point sessions)	PD (11,0)	1602
ANIHS9	Number of one-hop search requests sent by the network node	PD (11,0)	1608
ANSS19	Number of searches satisfied by the network node by sending one-hop search requests	PD (11,0)	1614
ANDSS9	Number of directed searches sent	PD (11,0)	1620
ASSDS9	Number of searches that were satisfied by sending directed searches	PD (11,0)	1626
ATDSR9	Cumulative elapsed time for directed search responses to be received	PD (11,0)	1632
ANDBE9	Number of domain broadcasts that have been run	PD (11,0)	1638
ANNDB9	Number of nodes that these domain broadcasts have been sent to	PD (11,0)	1644
ATRDB9	Cumulative elapsed time for the first positive response to be returned on domain broadcasts	PD (11,0)	1650
ATLRD9	Cumulative elapsed time for the last response to be returned on domain broadcasts	PD (11,0)	1656
ASSDB9	Number of searches that were satisfied by sending a domain broadcast	PD (11,0)	1662
ANBSE9	Number of broadcast searches that have been run	PD (11,0)	1668
ANNBS9	Number of adjacent nodes that these broadcast searches have been sent to	PD (11,0)	1674
ATRBS9	Cumulative elapsed time for the first positive response to be returned on broadcast searches	PD (11,0)	1680
ATLRB9	Cumulative elapsed time for the last response to be returned on broadcast searches	PD (11,0)	1686
ANSBS9	Number of searches that were satisfied by sending a broadcast search	PD (11,0)	1692
ATSPR9	Cumulative elapsed time from the start of search processing on the local node until a response has been returned to the local system to allow bind processing to continue	PD (11,0)	1698
ATSPC9	Cumulative elapsed time from the start of search processing until the local directory services task has completed all processing for the request. This measurement takes into account the time required to process domain broadcast or broadcast search responses even though a positive response has already been sent back to the local system to allow bind processing to continue	PD (11,0)	1704
ANRRT9	Number of request-route requests made to topology routing services (TRS)	PD (11,0)	1710
ASRRT9	Number of request-route requests made to TRS that were successful	PD (11,0)	1716
ATRRT9	Cumulative elapsed time for processing request-route requests	PD (11,0)	1722
AARRM9	Number of activate-route requests made to machine services control point (MSCP)	PD (11,0)	1728
AARCV9	Number of activate-route requests that require a controller description to be automatically created and/or varied on by the system	PD (11,0)	1734
ATRCV9	Cumulative elapsed time for automatic creation and/or vary on of the controller to be processed	PD (11,0)	1740
ASARR9	Number of successful activate-route requests processed by MSCP	PD (11,0)	1746

Figure A-31 (Page 13 of 13). APPN Data (One Entry per Interval)

Field Name	Description	Attributes	Buffer Position
ATARP9	Cumulative elapsed time for processing activate-route requests by MSCP	PD (11,0)	1752
	Network node processing a received-bind request from a node in the AS/400 network with routing information		
ANWAPA	Total number of work activities of this type processed	PD (11,0)	1758
ATWAPA	Cumulative elapsed time to complete work activities of this type	PD (11,0)	1764
ATWASA	Total number of work activities of this type that yielded a successful result	PD (11,0)	1770
ASPSPA	Number of session setup requests that are placed in pending status due to another session setup being in progress for the same local location, remote location, and mode triplet	PD (11,0)	1776
AARRMA	Number of activate-route requests made to machine services control point (MSCP)	PD (11,0)	1782
AARCVA	Number of activate-route requests that require a controller description to be automatically created and/or varied on by the system	PD (11,0)	1788
ATRCVA	Cumulative elapsed time for automatic creation and/or vary on of the controller to be processed	PD (11,0)	1794
ASARRA	Number of successful activate-route requests processed by MSCP	PD (11,0)	1800
ATARPA	Cumulative elapsed time for processing activate-route requests by MSCP	PD (11,0)	1806
	Network node processing a received-bind request from a node in a non-AS/400 network with routing information		
ANWAPB	Total number of work activities of this type processed	PD (11,0)	1812
ATWAPB	Cumulative elapsed time to complete work activities of this type	PD (11,0)	1818
ATWASB	Total number of work activities of this type that yielded a successful result	PD (11,0)	1824
ASPSPB	Number of session setup requests that are pended by another session setup in progress for the same local location, remote location, and mode	PD (11,0)	1830
AARRMB	Number of activate-route requests made to machine services control point (MSCP)	PD (11,0)	1836
AARCVB	Number of activate-route requests that require a controller description to be automatically created and/or varied on by the system	PD (11,0)	1842
ATRCVB	Cumulative elapsed time for automatic creation and/or vary on of the controller to be processed	PD (11,0)	1848
ASARRB	Number of successful activate-route requests processed by MSCP	PD (11,0)	1854
ATARPB	Cumulative elapsed time for processing activate-route requests by MSCP	PD (11,0)	1860

Appendix B. IBM-Supplied Object Contents

This appendix contains the contents of most of the IBM-supplied objects with the object types of Class (CLS), Job Description (JOBD), Job Queue (JOBQ), and Subsystem Description (SBSD). For each object, this figure provides:

- Object name
- Object description
- Command used to create the object
- Object parameters other than the default parameters

The objects are grouped by object type.

Figure B-1 (Page 1 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
Classes (CLS)			
QBATCH	Default batch job class	CRTCLS	CLS(QGPL/QBATCH) TIMESLICE(5000) PURGE(*NO) DFTWAIT(120) TEXT('Batch Subsystem Class')
QCTL	Controlling subsystem class	CRTCLS	CLS(QSYS/QCTL) RUNPTY(10) TIMESLICE(2000) DFTWAIT(30) TEXT('Controlling Subsystem Class')
QDIALOCAL	Class for QDIALOCAL job owned by QPGMR	CRTCLS	CLS(QGPL/QDIALOCAL) RUNPTY(20) TIMESLICE(2000) DFTWAIT(30) TEXT('Class for QDIALOCAL Job')
QDSNX	DSNX class	CRTCLS	CLS(QGPL/QDSNX) RUNPTY(50) DFTWAIT(300) AUT(*EXCLUDE) TEXT('DSNX Subsystem Class')
QFNC	Finance subsystem class	CRTCLS	CLS(QGPL/QFNC) RUNPTY(20) TIMESLICE(2000) DFTWAIT(30) TEXT('Finance Subsystem Class')
QINDUSR	Indirect user job class	CRTCLS	CLS(QGPL/QINDUSR) RUNPTY(50) TIMESLICE(5000) DFTWAIT(300) PURGE(*NO) TEXT('Class for Indirect User Job')
QINTER	Interactive subsystem class	CRTCLS	CLS(QGPL/QINTER) RUNPTY(20) TIMESLICE(2000) DFTWAIT(30) TEXT('Interactive Subsystem Class')

Figure B-1 (Page 2 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
QLPINSTALL	Licensed program installation class	CRTCLS	CLS(QSYS/QLPINSTALL) PURGE(*NO) DFTWAIT(120) AUT(*EXCLUDE) TEXT('Class for LP Install')
QMAILP	Host mail printing class	CRTCLS	CLS(QGPL/QMAILP) TIMESLICE(5000) PURGE(*NO) DFTWAIT(120) TEXT('Host Mail Printing Class')
QPGMR	Programmer sub-system class	CRTCLS	CLS(QGPL/QPGMR) RUNPTY(30) TIMESLICE(5000) DFTWAIT(30) TEXT('Programmer Subsystem Class')
QSNADS	SNADS class	CRTCLS	CLS(QGPL/QSNADS) RUNPTY(40) TIMESLICE(5000) DFTWAIT(30) TEXT('SNADS Subsystem Class')
QSPL	Spooling subsystem class	CRTCLS	CLS(QGPL/QSPL) RUNPTY(15) PURGE(*YES) DFTWAIT(30) TEXT('Spooling Subsystem Class')
QSPL2	Spooling subsystem class	CRTCLS	CLS(QGPL/QSPL2) RUNPTY(40) DFTWAIT(30) TEXT('Spooling Subsystem Class')
QSPL3	Spooling subsystem class	CRTCLS	CLS(QGPL/QSPL3) RUNPTY(30) DFTWAIT(30) TEXT('Spooling Subsystem Class')
QSYSCLS	Backup subsystem class	CRTCLS	CLS(QSYS/QSYSCLS) RUNPTY(10) TIMESLICE(2000) DFTWAIT(30) TEXT('Backup Subsystem Class')
QSYSCLS07	System subsystem class	CRTCLS	CLS(QSYS/QSYSCLS07) RUNPTY(7) TIMESLICE(2000) PURGE(*YES) DFTWAIT(30) CPUTIME(*NOMAX) MAXTMPSTG(*NOMAX) AUT(*USE) TEXT('System subsystem class with run priority 7'

Figure B-1 (Page 3 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
QSYSCLS10	System subsystem class	CRTCLS	CLS(QSYS/QSYSCLS10) RUNPTY(10) TIMESLICE(2000) PURGE(*YES) DFTWAIT(30) CPUTIME(*NOMAX) MAXTMPSTG(*NOMAX) AUT(*USE) TEXT('System subsystem class with run priority 10'
QSYSCLS20	System subsystem class	CRTCLS	CLS(QSYS/QSYSCLS20) RUNPTY(20) TIMESLICE(2000) PURGE(*YES) DFTWAIT(30) CPUTIME(*NOMAX) MAXTMPSTG(*NOMAX) AUT(*USE) TEXT('System subsystem class with run priority 20')
QSYSCLS35	System subsystem class	CRTCLS	CLS(QSYS/QSYSCLS35) RUNPTY(35) TIMESLICE(2000) PURGE(*YES) DFTWAIT(30) CPUTIME(*NOMAX) MAXTMPSTG(*NOMAX) AUT(*USE) TEXT('System subsystem class with run priority 35')
QSYSCLS50	System subsystem class	CRTCLS	CLS(QSYS/QSYSCLS50) RUNPTY(50) TIMESLICE(2000) PURGE(*YES) DFTWAIT(30) CPUTIME(*NOMAX) MAXTMPSTG(*NOMAX) AUT(*USE) TEXT('System subsystem class with run priority 50')
QWCPCSUP	PC support class	CRTCLS	CLS(QGPL/QWCPCSUP) RUNPTY(20) TIMESLICE(500) DFTWAIT(30) TEXT('PC Support Class')
Job Descriptions (JOBD)			
QBATCH	Batch subsystem job description	CRTJOBD	JOBD(QGPL/QBATCH) USER(QPGMR) LOG(4 0 *NOLIST) RTGDTA(QCMDB) TEXT('Batch Subsystem Job Description')

Figure B-1 (Page 4 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
QCTL	Controlling subsystem job description	CRTJOBD	JOBD(QSYS/QCTL) USER(*RQD) LOG(4 0 *NOLIST) TEXT('Controlling Subsystem Job Description')
QCTLIJBD	Controlling subsystem IGC job description	CRTJOBD	JOBD(QSYS/QCTLIJBD) USER(*RQD) RTGDTA(QIGC) LOG(4 0 *NOLIST) TEXT('Controlling Subsystem IGC JOBD')
QDFTJOBD	Default job description	CRTJOBD	JOBD(QGPL/QDFTJOBD) USER(*RQD) RTGDTA(QCMDI) AUT(*USE) LOG(4 0 *NOLIST) TEXT('Default Job Description')
QDIA	Document inter-change transaction program job description	CRTJOBD	JOBD(QGPL/QDIA) USER(QSNADS) JOBQ(QGPL/QSNADS) RTGDTA('QDIATP') LOG(4 0 *NOLIST) TEXT('DIA Transaction Program')
QDSNX	DSNX job description	CRTJOBD	JOBD(QGPL/QDSNX) USER(QDSNX) LOG(2 0 *MSG) RTGDTA(QDSNX) AUT(*EXCLUDE) TEXT('DSNX Subsystem Job Description')
QFNC	Finance subsystem job description	CRTJOBD	JOBD(QGPL/QFNC) USER(QFNC) JOBQ(QGPL/QFNC) RTGDTA(QFNC) LOG(4 0 *NOLIST) AUT(*EXCLUDE) TEXT('Finance Subsystem Job Description')
QHOSTPRT	Host printer job description	CRTJOBD	JOBD(QGPL/QHOSTPRT) USER(QPGMR) LOG(4 0 *NOLIST) TEXT('Host Printer Job Description')
QINTIJBD	Interactive subsystem IGC job description	CRTJOBD	JOBD(QGPL/QINTIJBD) RTGDTA(QIGC) LOG(4 0 *NOLIST) TEXT('Interactive Subsystem IGC JOBD')
QINTER	Interactive subsystem job description	CRTJOBD	JOBD(QGPL/QINTER) RTGDTA(QCMDI) LOG(4 0 *NOLIST) TEXT('Interactive Subsystem Job Description')

Figure B-1 (Page 5 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
QLPINSTALL	Licensed program installation job description	CRTJOBD	JOBD(QSYS/QLPINSTALL) USER(QLPINSTALL) JOBQ(QSYS/QLPINSTALL) RTGDTA(INIT) LOG(4 20 *NOLIST) AUT(*EXCLUDE) TEXT('Job description for LP Install')
QNFTP	Transaction program job description	CRTJOBD	JOBD(QGPL/QNFTP) USER(QSNADS) JOBQ(QGPL/QSNADS) RTGDTA('QNFTP') LOG(4 0 *NOLIST) TEXT('Transaction Program JOBD')
QPGMR	Job description used by QPGMR sub-system	CRTJOBD	JOBD(QGPL/QPGMR) USER(QPGMR) LOG(4 0 *NOLIST) AUT(*EXCLUDE) RTGDTA(QCMDI) TEXT('Programmer Job Description')
QPGMIJBD	Programmer sub-system IGC job description	CRTJOBD	JOBD(QGPL/QPGMIJBD) USER(QPGMR) LOG(4 0 *NOLIST) AUT(*EXCLUDE) RTGDTA(QIGC) TEXT('Programmer Subsystem IGC JOBD')
QSNADS	QSNADS subsystem job description	CRTJOBD	JOBD(QGPL/QSNADS) USER(QSNADS) RTGDTA(QSTARTUP) LOG(4 0 *NOLIST) JOBQ(QGPL/QSNADS) TEXT('QSNADS Subsystem Job Description')
QSPLAFPW	Advanced function print writer job description	CRTJOBD	JOBD(QGPL/QSPLAFPW) USER(QSPLJOB) JOBQ(QGPL/QSPL) RTGDTA(QAFPWT) LOG(4 0 *NOLIST) INQMSGRPY(*SYSRPYL) TEXT('Advanced Function Print Writer')
QSPLDBR	Database spooling reader job description	CRTJOBD	JOBD(QGPL/QSPLDBR) USER(QSPLJOB) JOBQ(QGPL/QSPL) RTGDTA(QRDRDB) LOG(4 0 *NOLIST) INQMSGRPY(*SYSRPYL) TEXT('Database Spooling Reader')
QSPLDKTR	Diskette spooling reader job description	CRTJOBD	JOBD(QGPL/QSPLDKTR) USER(QSPLJOB) JOBQ(QGPL/QSPL) RTGDTA(QRDRDK) LOG(4 0 *NOLIST) INQMSGRPY(*SYSRPYL) TEXT('Diskette Spooling Reader')

Figure B-1 (Page 6 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
QSPLDKTW	Diskette spooling writer job description	CRTJOBD	JOBD(QGPL/QSPLDKTW) USER(QSPLJOB) JOBQ(QGPL/QSPL) RTGDTA(QWTRDK) LOG(4 0 *NOLIST) INQMSGRPY(*SYSRPYL) TEXT('Diskette Spooling Writer')
QSPLERROR	Spooling error job description	CRTJOBD	JOBD(QSYS/QSPLERROR) USER(QSPLJOB) JOBQ(QGPL/QBATCH) LOG(1 0 *SECLVL) TEXT('Error Job Use')
QSPLPRTW	Printer spooling writer job description	CRTJOBD	JOBD(QGPL/QSPLPRTW) USER(QSPLJOB) JOBQ(QGPL/QSPL) RTGDTA(QWTRPT) LOG(4 0 *NOLIST) INQMSGRPY(*SYSRPYL) TEXT('Printer Spooling Writer')
QSTRUPJD	Autostart job description	CRTJOBD	JOBD(QSYS/QSTRUPJD) USER(QPGMR) LOG(4 10 *SECLVL) AUT(*EXCLUDE) RQSDTA('CALL QSYS/QWDAJPGM') TEXT('Autostart Job JOBD')
QSYSJOBD	Backup job description	CRTJOBD	JOBD(QSYS/QSYSJOBD) USER(*RQD) RTGDTA(QCMDI) LOG(4 0 *NOLIST) AUT(*USE) TEXT('Backup Job Description')
QSYSWRK	System subsystem job description	CRTJOBD	JOBD(QSYS/QSYSWRK) USER(QPGMR) LOG(4 10 *SECLVL) RQSDTA('CALL QSYS/QWDSYWRK') AUT(*USE) TEXT('System subsystem JOBD')
QS36MRT	Multiple requestor terminal job description	CRTJOBD	JOBD(QGPL/QS36MRT) USER(QPGMR) LOG(4 0 *NOLIST) AUT(*USE) TEXT('MRT Job Description')
Job Queues (JOBQ)			
QBASE	QBASE subsystem job queue	CRTJOBQ	JOBQ(QGPL/QBASE) TEXT('QBASE Subsystem Job Queue')
QBATCH	Batch subsystem job queue	CRTJOBQ	JOBQ(QGPL/QBATCH) TEXT(Batch Subsystem Job Queue')
QCTL	Controlling subsystem job queue	CRTJOBQ	JOBQ(QSYS/QCTL) TEXT('Controlling Subsystem Job Queue')

Figure B-1 (Page 7 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
QFNC	Finance subsystem job queue	CRTJOBQ	JOBQ(QGPL/QFNC) TEXT('Finance Subsystem Job Queue')
QINTER	Interactive subsystem job queue	CRTJOBQ	JOBQ(QGPL/QINTER) TEXT('Interactive Subsystem Job Queue')
QLPINSTALL	Licensed program installation job queue	CRTJOBQ	JOBQ(QSYS/QLPINSTALL) AUT(*EXCLUDE) TEXT('Job Queue for LP Install')
QPGMR	Programmer sub-system job queue	CRTJOBQ	JOBQ(QGPL/QPGMR) TEXT('Programmer Subsystem Job Queue')
QSNADS	QSNADS subsystem job queue	CRTJOBQ	JOBQ(QGPL/QSNADS) TEXT('QSNADS Subsystem Job Queue')
QSPL	Spooling subsystem job queue	CRTJOBQ	JOBQ(QGPL/QSPL) TEXT('Spooling Subsystem Job Queue')
QNMSVQ	SystemView* server job queue	CRTJOBQ	JOBQ(QSYS/QNMSVQ) OPRCTL(*YES) AUT(*EXCLUDE) TEXT('Job Queue for SystemView Server Jobs')
QSYSNOMAX	QSYSNOMAX system subsystem job queue	CRTJOBQ	JOBQ(QSYS/QSYSNOMAX) OPRCTL(*YES) AUT(*EXCLUDE) TEXT('System Subsystem Job Queue')
QSYSSBSD	QSYSSBSD sub-system job queue	CRTJOBQ	JOBQ(QSYS/QSYSSBSD) TEXT('QSYSSBSD Subsystem Job Queue')
QS36EVOKE	QS36EVOKE job queue	CRTJOBQ	JOBQ(QGPL/QS36EVOKE) TEXT('QS36EVOKE Job Queue')
QS36MRT	QS36 job queue	CRTJOBQ	JOBQ(QGPL/QS36MRT) TEXT('QS36MRT Job Queue')
QTXTSRCH	Text Search Services job queue	CRTJOBQ	JOBQ(QGPL/QTXTSRCH) TEXT('Text Search Services Job Queue')
Subsystem Descriptions (SBSD)			
QBASE	Basic controlling sub-system description	CRTSBSD	SBSD(QSYS/QBASE) POOLS((1 *BASE) (2 *INTERACT)) AUT(*CHANGE) TEXT('Basic Controlling Subsystem')
		ADDJOBQE	SBSD(QSYS/QBASE) JOBQ(QGPL/QBATCH) SEQNBR(10)
		ADDJOBQE	SBSD(QSYS/QBASE) JOBQ(QGPL/QINTER) MAXACT(*NOMAX) SEQNBR(50)
		ADDJOBQE	SBSD(QSYS/QBASE) JOBQ(QSYS/QCTL) MAXACT(*NOMAX) SEQNBR(100)

Figure B-1 (Page 8 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDJOBQE	SBSD(QSYS/QBASE) JOBQ(QGPL/QS36MRT) MAXACT(*NOMAX) SEQNBR(150)
		ADDJOBQE	SBSD(QSYS/QBASE) JOBQ(QGPL/QS36EVOKE) MAXACT(*NOMAX) SEQNBR(200)
		ADDJOBQE	SBSD(QSYS/QBASE) JOBQ(QGPL/QBASE) MAXACT(*NOMAX) SEQNBR(250)
		ADDWSE	SBSD(QSYS/QBASE) WRKSTNTYPE(*CONS)
		ADDWSE	SBSD(QSYS/QBASE) WRKSTNTYPE(*ALL)
		ADDWSE	SBSD(QSYS/QBASE) WRKSTNTYPE(5555) /* IGC ONLY */
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(10) CMPVAL(QCMDB) PGM(QSYS/QCMD) CLS(QGPL/QBATCH)
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(50) CMPVAL(QCMDI) PGM(QSYS/QCMD) CLS(QGPL/QINTER) POOLID(2)
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(100) CMPVAL(QS36EVOKE) PGM(QSYS/QCMD) CLS(QGPL/QBATCH)
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(150) CMPVAL(QS36MRT) PGM(QSYS/QCMD) CLS(QGPL/QINTER) POOLID(2)
		ADDRTGE	SBSD(QSYS/QBASE) /* IGC ONLY */ SEQNBR(200) CMPVAL((QIGC) PGM(QSYS/QCMD) CLS(QGPL/QINTER) POOLID(2)

Figure B-1 (Page 9 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(300) CMPVAL(525XTEST) PGM(QSYS/QARDRIVE) CLS(QGPL/QINTER) POOLID(2)
		ADDRTGE	SBSD(QSYS/QBASE) /* BEFORE */ SEQNBR(350) /* PGMEVOKE */ CMPVAL(QVPPRINT 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QBASE) /* BEFORE */ SEQNBR(400) /* PGMEVOKE */ CMPVAL(QTFDWNLD 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QBASE) /* BEFORE */ SEQNBR(450) /* PGMEVOKE */ CMPVAL(QMFRCVR 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QBASE) /* BEFORE */ SEQNBR(500) /* PGMEVOKE */ CMPVAL(QMFSNDR 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(510) CMPVAL(QHQTRGT 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(520) CMPVAL(QRQSRV 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(530) CMPVAL(QPCSUPP 37) PGM(*RTGDTA) CLS(QPGL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QBASE) /* BEFORE */ SEQNBR(550) /* PGMEVOKE */ CMPVAL(QCNPCSUP 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(600) CMPVAL(PGMEVOKE 29) PGM(*RTGDTA) CLS(QGPL/QBATCH)

Figure B-1 (Page 10 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(650) CMPVAL(QCMD38) PGM(QSYS/QCL) CLS(QGPL/QINTER) POOLID(2)
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(9999) CMPVAL(*ANY) PGM(QSYS/QCMD) CLS(QGPL/QBATCH)
		ADDRTGE	SBSD(QSYS/QBASE) SEQNBR(275) CMPVAL(#INTER) PGM(*RTGDTA) CLS(QGPL/QINTER)
		ADDPJE	SBSD(QSYS/QBASE) PGM(QSYS/QOQSESRV) STRJOBS(*NO) INLJOBS(10) THRESHOLD(3) ADLJOBS(3) MAXJOBS(*NOMAX) JOB(QOQSESRV) JOBD(*USRPRF) MAXUSE(200) WAIT(*YES) POOLID(1) CLS(QGPL/QWCPCSUP *CALC *NONE 0)
		ADDCMNE	SBSD(QSYS/QBASE) DFTUSR(*SYS) DEV(*ALL)
		ADDCMNE	SBSD(QSYS/QBASE) DFTUSR(*NONE) MODE(QPCSUPP) DEV(*ALL)
		ADDAJE	SBSD(QSYS/QBASE) JOB(QSTRUPJD) JOBD(QSYS/QSTRUPJD)
		ADDAJE	SBSD(QSYS/QBASE) JOB(QPFRCOL) JOBD(QGPL/QPFRCOL)
QBATCH	Batch subsystem description	CRTSBSD	SBSD(QSYS/QBATCH) POOLS((1 *BASE)) MAXJOBS(*NOMAX) TEXT('Batch Subsystem')
		ADDJOBQE	SBSD(QSYS/QBATCH) JOBQ(QGPL/QBATCH) MAXACT(1)

Figure B-1 (Page 11 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDJOBQE	SBSD(QSYS/QBATCH) JOBQ(QGPL/QS36EVOKE) SEQNBR(20) MAXACT(*NOMAX)
		ADDJOBQE	SBSD(QSYS/QBATCH) JOBQ(QGPL/QTXTSRCH) SEQNBR(50) MAXACT(*NOMAX)
		ADDRTGE	SBSD(QSYS/QBATCH) /* IGC ONLY */ SEQNBR(15) CMPVAL(QIGC) CLS(QGPL/QBATCH) PGM(QSYS/QCMD)
		ADDRTGE	SBSD(QSYS/QBATCH) SEQNBR(300) CMPVAL(QS36EVOKE) CLS(QGPL/QBATCH) PGM(QSYS/QCMD)
		ADDRTGE	SBSD(QSYS/QBATCH) SEQNBR(700) CMPVAL(QCMD38) PGM(QSYS/QCL) CLS(QGPL/QBATCH)
		ADDRTGE	SBSD(QSYS/QBATCH) SEQNBR(9999) CMPVAL(*ANY) PGM(QSYS/QCMD) CLS(QGPL/QBATCH)
QCMN	Communications sub-system description	CRTSBSD	SBSD(QSYS/QCMN) POOLS((1 *BASE)) MAXJOBS(*NOMAX) TEXT('Communications Subsystem')
		ADDRTGE	SBSD(QSYS/QCMN) /* BEFORE */ SEQNBR(50) /* PGMEVOKE */ CMPVAL(QVPPRINT 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QCMN) /* BEFORE */ SEQNBR(100) /* PGMEVOKE */ CMPVAL(QTFDWNLD 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QCMN) /* BEFORE */ SEQNBR(150) /* PGMEVOKE */ CMPVAL(QMFRCVR 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)

Figure B-1 (Page 12 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDRTGE	SBSD(QSYS/QCMN) /* BEFORE */ SEQNBR(200) /* PGMEVOKE */ CMPVAL(QMFSNDR 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QCMN) SEQNBR(210) CMPVAL(QHQTRGT 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QCMN) SEQNBR(220) CMPVAL(QRQSRV 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QCMN) SEQNBR(230) CMPVAL(QPCSUPP 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QCMN) /* BEFORE */ SEQNBR(250) /* PGMEVOKE */ CMPVAL(QCNPCSUP 37) PGM(*RTGDTA) CLS(QGPL/QWCPCSUP)
		ADDRTGE	SBSD(QSYS/QCMN) SEQNBR(275) CMPVAL(#INTER) PGM(*RTGDTA) CLS(QGPL/QINTER)
		ADDRTGE	SBSD(QSYS/QCMN) SEQNBR(300) CMPVAL(PGMEVOKE 29) PGM(*RTGDTA) CLS(QGPL/QBATCH)
		ADDCMNE	SBSD(QSYS/QCMN) DFTUSR(*SYS) DEV(*ALL)
		ADDCMNE	SBSD(QSYS/QCMN) DFTUSR(*NONE) MODE(QPCSUPP) DEV(*ALL)

Figure B-1 (Page 13 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDPJE	SBSD(QSYS/QCMN) PGM(QSYS/QOQSESRV) STRJOBS(*NO) INLJOBS(10) THRESHOLD(3) ADLJOBS(3) MAXJOBS(*NOMAX) JOB(QOQSESRV) JOBD(*USRPRF) MAXUSE(200) WAIT(*YES) POOLID(1) CLS(QGPL/QWCPCSUP *CALC *NONE 0)
QCTL	Controlling subsystem description	CRTSBSD	SBSD(QSYS/QCTL) POOLS((1 *BASE)) TEXT('Controlling Subsystem')
		ADDJOBQE	SBSD(QSYS/QCTL) JOBQ(QSYS/QCTL) MAXACT(*NOMAX)
		ADDWSE	SBSD(QSYS/QCTL) WRKSTNTYPE(*CONS)
		ADDWSE	SBSD(QSYS/QCTL) WRKSTNTYPE(*ALL) AT(*ENTER)
		ADDWSE	SBSD(QSYS/QCTL) /* For IGC */ WRKSTNTYPE(5555) AT(*ENTER)
		ADDRTGE	SBSD(QSYS/QCTL) SEQNBR(10) CMPVAL(525XTEST) PGM(QSYS/QARDRIVE) CLS(QSYS/QCTL)
		ADDRTGE	SBSD(QSYS/QCTL) /* IGC ONLY */ SEQNBR(15) CMPVAL(QIGC) PGM(QSYS/QCMD) CLS(QSYS/QCTL)
		ADDRTGE	SBSD(QSYS/QCTL) SEQNBR(700) CMPVAL(QCMD38) PGM(QSYS/QCL) CLS(QSYS/QCTL)
		ADDRTGE	SBSD(QSYS/QCTL) SEQNBR(9999) CMPVAL(*ANY) PGM(QSYS/QCMD) CLS(QSYS/QCTL)
		ADDAJE	SBSD(QSYS/QCTL) JOB(QSTRUPJD) JOBD(QSYS/QSTRUPJD)

Figure B-1 (Page 14 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDAJE	SBSD(QSYS/QCTL) JOB(QPFRCOL) JOBD(QGPL/QPFRCOL)
QDSNX	DSNX subsystem description	CRTSBSD	SBSD(QGPL/QDSNX) POOLS((1 *BASE)) AUT(*EXCLUDE) TEXT('DSNX Subsystem Description')
		ADDRTGE	SBSD(QGPL/QDSNX) SEQNBR(10) CMPVAL(QDSNX) PGM(QSYS/QDXDDOER) CLS(QGPL/QDSNX)
		ADDAJE	SBSD(QGPL/QDSNX) JOB(QDSNX) JOBD(QGPL/QDSNX)
QFNC	Finance subsystem description	CRTSBSD	SBSD(QGPL/QFNC) POOLS((1 *BASE)) MAXJOBS(*NOMAX) TEXT('Finance Subsystem')
		ADDJOBQE	SBSD(QGPL/QFNC) JOBQ(QGPL/QFNC) SEQNBR(10) MAXACT(*NOMAX)
		ADDRTGE	SBSD(QGPL/QFNC) SEQNBR(10) CMPVAL(QFNC) PGM(QSYS/QCMD) CLS(QGPL/QFNC) POOLID(1)
		ADDRTGE	SBSD(QGPL/QFNC) SEQNBR(20) CMPVAL(QCMD38) PGM(QSYS/QCL) CLS(QGPL/QFNC) POOLID(1)
QINTER	Interactive subsystem description	CRTSBSD	SBSD(QSYS/QINTER) POOLS((1 *BASE)(2 *INTERACT)) TEXT('Interactive Subsystem')
		ADDJOBQE	SBSD(QSYS/QINTER) JOBQ(QGPL/QINTER) MAXACT(*NOMAX)
		ADDJOBQE	SBSD(QSYS/QINTER) JOBQ(QGPL/QS36MRT) SEQNBR(20) MAXACT(*NOMAX)
		ADDWSE	SBSD(QSYS/QINTER) WRKSTNTYPE(*ALL)
		ADDWSE	SBSD(QSYS/QINTER) /* IGC ONLY */ WRKSTNTYPE(5555)

Figure B-1 (Page 15 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDWSE	SBSD(QSYS/QINTER) WRKSTNTYPE(*CONS) AT(*ENTER)
		ADDRTGE	SBSD(QSYS/QINTER) SEQNBR(10) CMPVAL(QCMDI) PGM(QSYS/QCMD) POOLID(2) CLS(QGPL/QINTER)
		ADDRTGE	SBSD(QSYS/QINTER) /* IGC ONLY */ SEQNBR(15) CMPVAL(QIGC) PGM(QSYS/QCMD) POOLID(2) CLS(QGPL/QINTER)
		ADDRTGE	SBSD(QSYS/QINTER) SEQNBR(20) CMPVAL(QS36MRT) PGM(QSYS/QCMD) POOLID(2) CLS(QGPL/QINTER)
		ADDRTGE	SBSD(QSYS/QINTER) SEQNBR(40) CMPVAL(525XTEST) PGM(QSYS/QARDRIVE) POOLID(2) CLS(QGPL/QINTER)
		ADDRTGE	SBSD(QSYS/QINTER) SEQNBR(700) CMPVAL(QCMD38) PGM(QSYS/QCL) POOLID(2) CLS(QGPL/QINTER)
		ADDRTGE	SBSD(QGPL/QINTER) SEQNBR(9999) CMPVAL(*ANY) PGM(QSYS/QCMD) POOLID(2) CLS(QGPL/QINTER)
QLPINSTALL	Licensed program installation subsystem description	CRTSBSD	SBSD(QSYS/QLPINSTALL) POOLS((1 *BASE)) AUT(*EXCLUDE) TEXT('Subsystem for LP Install')
		ADDJOBQE	SBSD(QSYS/QLPINSTALL) JOBQ(QSYS/QLPINSTALL) MAXACT(*NOMAX)
		ADDRTGE	SBSD(QSYS/QLPINSTALL) SEQNBR(10) CMPVAL(INIT) PGM(QSYS/QLPCTLIN) CLS(QSYS/QLPINSTALL)

Figure B-1 (Page 16 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDRTGE	SBSD(QSYS/QLPINSTALL) SEQNBR(20) CMPVAL(BATCH) PGM(QSYS/QCMD) CLS(QSYS/QLPINSTALL)
		ADDAJE	SBSD(QSYS/QLPINSTALL) JOB(QLPINSTALL) JOBD(QSYS/QLPINSTALL)
QPGMR	Programmer sub-system description	CRTSBSD	SBSD(QSYS/QPGMR) POOLS((1 *BASE)(2 *BASE)) TEXT('Programmer Subsystem')
		ADDJOBQE	SBSD(QSYS/QPGMR) JOBQ(QGPL/QPGMR)
		ADDWSE	SBSD(QSYS/QPGMR) WRKSTNTYPE(*ALL) AT(*ENTER)
		ADDWSE	SBSD(QSYS/QPGMR) /* IGC ONLY */ WRKSTNTYPE(5555) AT(*ENTER)
		ADDRTGE	SBSD(QSYS/QPGMR) SEQNBR(10) CMPVAL(525XTEST) PGM(QSYS/QARDRIVE) CLS(QGPL/QPGMR)
		ADDRTGE	SBSD(QSYS/QPGMR) /* IGC ONLY */ SEQNBR(15) CMPVAL(QIGC) PGM(QSYS/QCMD) CLS(QGPL/QPGMR)
		ADDRTGE	SBSD(QSYS/QPGMR) SEQNBR(20) CMPVAL(QCMDB) PGM(QSYS/QCMD) CLS(QGPL/QBATCH)
		ADDRTGE	SBSD(QSYS/QPGMR) SEQNBR(700) CMPVAL(QCMD38) PGM(QSYS/QCL) CLS(QGPL/QBATCH)
		ADDRTGE	SBSD(QSYS/QPGMR) SEQNBR(9999) CMPVAL(*ANY) PGM(QSYS/QCMD) CLS(QGPL/QPGMR) POOLID(2)
QSNADS	SNA distribution sub-system description	CRTSBSD	SBSD(QSYS/QSNADS) POOLS((1 *BASE)) MAXJOBS(*NOMAX) TEXT('SNA Distributions Subsystem')

Figure B-1 (Page 17 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDAJE	SBSD(QSYS/QSNADS) JOB(QZDSTART) JOBD(QGPL/QSNADS)
		ADDJOBQE	SBSD(QSYS/QSNADS) JOBQ(QGPL/QSNADS) MAXACT(*NOMAX)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(100) CMPVAL(QSTARTUP) CLS(QGPL/QSNADS) PGM(QSYS/QZDSTRUP) MAXACT(1)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(200) CMPVAL(QROUTER) CLS(QGPL/QSNADS) PGM(QSYS/QZDROUTER) MAXACT(1)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(275) CMPVAL(#INTER) PGM(*RTGDTA) CLS(QGPL/QINTER)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(300) CMPVAL(QSENDER) CLS(QGPL/QSNADS) PGM(QSYS/QZDSTSND) MAXACT(128)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(350) CMPVAL(QSVDSSND) CLS(QGPL/QSNADS) PGM(QS2STSND)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(500) CMPVAL(PGMEVOKE 29) PGM(*RTGDTA) CLS(QGPL/QSNADS) MAXACT(*NOMAX)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(600) CMPVAL(QDIATP) GM(QSYS/QOSDIATP) CLS(QGPL/QSNADS) MAXACT(1)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(650) CMPVAL(QDIALOCAL) PGM(QSYS/QOSASYNC) CLS(QGPL/QDIALOCAL) MAXACT(2)

Figure B-1 (Page 18 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(660) CMPVAL(QDIAINDUSR) PGM(QSYS/QOSMLFMT) CLS(QGPL/QINDUSR) MAXACT(2)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(700) CMPVAL(QNFTP) PGM(QSYS/QNFTPDTA) CLS(QGPL/QSNADS) MAXACT(2)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(800) MPVAL(QGATEWAY) PGM(QSYS/QZDSTGAT) CLS(QGPL/QSNADS) MAXACT(*NOMAX)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(900) CMPVAL(RPDSLINE) PGM(QRJE/QGTPUTCL) CLS(QGPL/QSNADS) MAXACT(*NOMAX)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(1000) CMPVAL(RPDSRCVR) PGM(QRJE/QGTISNAD) CLS(QGPL/QSNADS) MAXACT(*NOMAX)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(1100) CMPVAL(QDIAHSTPRT) PGM(QSYS/QOHMAILP) CLS(QGPL/QMAILP) MAXACT(1)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(1200) CMPVAL(QDIANOTIFY) PGM(QSYS/QOHSTSL) CLS(QGPL/QBATCH) MAXACT(1)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(1300) CMPVAL(QESTP) PGM(QSYS/QESFXRCV) CLS(QGPL/QSNADS) MAXACT(1)

Figure B-1 (Page 19 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(1400) CMPVAL('QX4SNMTA' 1) PGM(QX400/QX4SNMTA) CLS(QGPL/QSNADS) MAXACT(1) POOLID(1)
		ADDRTGE	SBSD(QSYS/QSNADS) SEQNBR(1500) CMPVAL('SMTPRTGD' 1) PGM(QTCP/QTMSTRBR) CLS(QGPL/QSNADS) MAXACT(*NOMAX) POOLID(1)
QSPL	Spooling subsystem description	CRTSBSD	SBSD(QSYS/QSPL) POOLS((1 *BASE)(2 *SPOOL)) MAXJOBS(*NOMAX) TEXT('Spooling Subsystem')
		ADDJOBQE	SBSD(QSYS/QSPL) JOBQ(QGPL/QSPL) MAXACT(*NOMAX)
		ADDRTGE	SBSD(QSYS/QSPL) SEQNBR(10) CMPVAL(QWTRPT) PGM(QSYS/QCMD) CLS(QGPL/QSPL) POOLID(2)
		ADDRTGE	SBSD(QSYS/QSPL) SEQNBR(50) CMPVAL(QAFPWT) PGM(QSYS/QCMD) CLS(QGPL/QSPL3) POOLID(2)
		ADDRTGE	SBSD(QSYS/QSPL) SEQNBR(9999) CMPVAL(*ANY) PGM(QSYS/QCMD) CLS(QGPL/QSPL2)
QSYSSBSD	Backup controlling subsystem description	CRTSBSD	SBSD(QSYS/QSYSSBSD) POOLS((1 *BASE)) AUT(*USE) TEXT('Backup Controlling Subsystem')
		ADDJOBQE	SBSD(QSYS/QSYSSBSD) JOBQ(QSYS/QSYSSBSD) MAXACT(*NOMAX) SEQNBR(10)
		ADDWSE	SBSD(QSYS/QSYSSBSD) WRKSTNTYPE(*CONS) JOBD(QSYS/QSYSJOBD)

Figure B-1 (Page 20 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDRTGE	SBSD(QSYS/QSYSSBSD) SEQNBR(400) CMPVAL(525XTEST) PGM(QARDRIVE) CLS(QSYS/QSYSCLS)
		ADDRTGE	SBSD(QSYS/QSYSSBSD) SEQNBR(500) CMPVAL(QCMD38) PGM(QSYS/QCL) CLS(QSYS/QSYSCLS)
		ADDRTGE	SBSD(QSYS/QSYSSBSD) SEQNBR(9999) CMPVAL(*ANY) PGM(QSYS/QCMD) CLS(QSYS/QSYSCLS)
QSYSWRK	System subsystem description	CRTSBSD	SBSD(QSYS/QSYSWRK) POOLS((1 *BASE)) MAXJOBS(*NOMAX) SGNDSPF(*QDSIGNON) AUT(*CHANGE) TEXT('System subsystem')
		ADDAJE	SBSD(QSYS/QSYSWRK) JOB(QSYSWRKJOB) JOBD(QSYS/QSYSWRK)
		ADDRTGE	SBSD(QSYS/QSYSWRK) SEQNBR(10) CMPVAL(RUNPTY07) PGM(QSYS/QCMD) CLS(QSYS/QSYSCLS07) MAXACT(*NOMAX) POOLID(1)
		ADDRTGE	SBSD(QSYS/QSYSWRK) SEQNBR(20) CMPVAL(RUNPTY10) PGM(QSYS/QCMD) CLS(QSYS/QSYSCLS10) MAXACT(*NOMAX) POOLID(1)
		ADDRTGE	SBSD(QSYS/QSYSWRK) SEQNBR(30) CMPVAL(RUNPTY20) PGM(QSYS/QCMD) CLS(QSYS/QSYSCLS20) MAXACT(*NOMAX) POOLID(1)
		ADDRTGE	SBSD(QSYS/QSYSWRK) SEQNBR(40) CMPVAL(RUNPTY35) PGM(QSYS/QCMD) CLS(QSYS/QSYSCLS35) MAXACT(*NOMAX) POOLID(1)

Figure B-1 (Page 21 of 21). Contents of IBM-Supplied Objects

Object	Description	Command	Parameters Other Than Default
		ADDRTGE	SBSD(QSYS/QSYSWRK) SEQNBR(50) CMPVAL(RUNPTY50) PGM(QSYS/QCMD) CLS(QSYS/QSYSCLS50) MAXACT(*NOMAX) POOLID(1)
		ADDRTGE	SBSD(QSYS/QSYSWRK) SEQNBR(9999) CMPVAL(*ANY) PGM(QSYS/QCMD) CLS(QSYS/QSYSCLS50) MAXACT(*NOMAX) POOLID(1)
		ADDJOBQE	SBSD(QSYS/QSYSWRK) JOBQ(QSYS/QSYSNOMAX) MAXACT(*NOMAX) SEQNBR(10)
		ADDJOBQE	SBSD(QSYS/QSYSWRK) JOBQ(QSYS/QNMSVQ) MAXACT(5) SEQNBR(40)

Figure B-2. Source for CL Start-up Program

Object	Command	CL Program Source
QSTRUP	CRTCLPGM	PGM DCL VAR(&STRWTRS) TYPE(*CHAR) LEN(1) DCL VAR(&CTLSBSD) TYPE(*CHAR) LEN(20) DCL VAR(&CPYR) TYPE(*CHAR) LEN(90) VALUE('+ 5738-SS1 (C) COPYRIGHT IBM CORP 1980 1993. + LICENSED MATERIAL - PROGRAM PROPERTY OF IBM') QSYS/STRSBS SBSD(QSPL) MONMSG MSGID(CPF0000) QSYS/RLSJOBQ JOBQ(QGPL/QS36MRT) MONMSG MSGID(CPF0000) QSYS/RLSJOBQ JOBQ(QGPL/QS36EVOKE) MONMSG MSGID(CPF0000) QSYS/STRCLNUP MONMSG MSGID(CPF0000) QSYS/RTVSYSVAL SYSVAL(QSTRPRTWTR) + RTNVAR(&STRWTRS) IF COND(&STRWTRS = '0') THEN (GOTO CMDLBL (NOWTRS)) QSYS/STRPRTWTR DEV(*ALL) MONMSG MSGID(CPF0000) NOWTRS: QSYS/RTVSYSVAL SYSVAL(QCTLSBSD) + RTNVAR(&CTLSBSD) IF (COND(&CTLSBSD *NE 'QCTL QSYS ') + *AND (&CTLSBSD *NE 'QCTL QGPL ')) THEN (GOTO CMDLBL(DONE)) QSYS/STRSBS SBSD(QINTER) MONMSG MSGID(CPF0000) QSYS/STRSBS SBSD(QBATCH) MONMSG MSGID(CPF0000) QSYS/STRSBS SBSD(QCMN) MONMSG MSGID(CPF0000) DONE: RETURN CHGVAR VAR(&CPYR) VALUE(&CPYR) /* Needed to include CPYR variable in pgm */ ENDPGM

Appendix C. Characteristics of the Shipped System

The system is shipped with a predefined set of work management objects that can be used immediately for processing. Refer to Appendix B, "IBM-Supplied Object Contents," for a detailed description of these objects. You can continue to use these objects or you can create your own.

The user profiles shipped with the system are designed for the various types of system users. For example, normal programmer functions are authorized to the QPGMR user profile; security officer functions are authorized to the QSECOFR user profile; and system operator functions are authorized to the QSYSOPR user profile. The user profile associated with a job determines what system functions can be done by that job.

These user profiles and subsystem descriptions are set up in order that any of several IBM-supplied programs such as QSYS/QCMD are called to support interactive jobs.

The program QSYS/QCMD processes CL commands specified interactively or in batch jobs. It is called during the running of most jobs that run on the system as it is shipped. An initial program can be specified in a user profile. If a job is routed to QSYS/QCMD and the user profile under which the job is running has an initial program, QSYS/QCMD calls the initial program. If no initial program is specified, QSYS/QCMD displays the initial menu.

Similarly, the system menu is displayed for the QSYSOPR user profile as the initial menu system.

The Controlling Subsystem as Shipped by IBM

The controlling subsystem is the subsystem description name specified in the system value QCTLSBSD. When your system is shipped, QSYS/QBASE is specified in QCTLSBSD as the controlling subsystem. QSYS/QBASE specifies that all work stations are available for sign-on. The console is the only work station that can be used during an attended IPL. Therefore, if you assign another subsystem as the controlling subsystem, you must have a work station entry for the console in that subsystem description. This work station entry should specify that the subsystem display the Sign On display (specify WRKSTNTYPE(*CONS) AT(*SIGNON) on the Add Work Station Entry (ADDWSE) command).

The IBM-Supplied QBASE and QCMN Subsystem Descriptions

Before program start requests are accepted by the system, a subsystem that supports communications must be started. There are two IBM-supplied subsystems, QBASE and QCMN, that accept program start requests for all communications types. QBASE is the default controlling subsystem and QCMN is the communications subsystem. Having either of these subsystems active allows program start requests to be accepted for all communications types.

The QCMN and QBASE subsystems have device type entries of *ALL and *ANY. Both subsystems have the appropriate routing entries (with CMPVAL(PGMEVOKE 29)) so all program start requests received by the system are accepted. If you have either of these subsystems and then start your own communications subsystem, or other subsystems such as DSNX(QDSNX) or SNADS(QSNADS), read the following section, "Allocating Communications Devices and Modes" on page 3-15. Both subsystems will be attempting to allocate the same communications devices.

Important System Values Shipped by IBM

The system is shipped with many system values that control different aspects of system operation. The following are some of the most significant ones (see Chapter 2, "System Values and Network Attributes," for a full description of all system values).

QAUTOCFG: Controls whether the system should create device descriptions automatically for devices such as work stations that you attach locally to your system. The default is to perform automatic configuration. If you do not want the system to do this for you, enter the following command:

```
CHGSYSVAL SYSVAL(QAUTOCFG) VALUE('0')
```

QCTLSBSD: Specifies the name of the controlling subsystem description. This is used to automatically start the controlling subsystem during every IPL. The default is to use the QBASE shipped subsystem description. You can change this to specify a different subsystem description, and it will be used on the next IPL. For example, if you want to use the QCTL shipped subsystem description for the controlling subsystem, enter the following command (see the next section for more information about the shipped subsystem descriptions):

```
CHGSYSVAL SYSVAL(QCTLSBSD) VALUE('QCTL QSYS')
```

QDEVNAMING: Specifies what locally attached device names are to be assigned whenever the system automatically creates device descriptions. This is primarily for automatic configuration, but also applies to the device description created for the console. The default is to use names like DSP01 for work stations and PRT01 for printers. If you want to have names like W1 for work stations and P1 for printers, enter the following command:

```
CHGSYSVAL SYSVAL(QDEVNAMING) VALUE(*S36)
```

QIPLTYPE: Specifies how IPL is to occur. The default is that the IPL will be unattended, meaning that once you start the IPL and the key is not in the MANUAL position, all the system activity will be started without any operator intervention. If you want the operator to have a chance to select any special functions during the IPL or change certain system characteristics, set the Keylock switch to Manual or enter the following command:

```
CHGSYSVAL SYSVAL(QIPLTYPE) VALUE('1')
```

This causes the IPL to be attended, meaning that the normal system activity is not started until after the operator has responded to a series of displays. If the Keylock switch is set to Manual, the IPL will be attended regardless of this system value.

QSECURITY: Specifies the type of security enforced by the system. The default is to not require passwords or authority. If you want passwords to be required and authority checking to be done, enter the following command:

```
CHGSYSVAL SYSVAL(QSECURITY) VALUE('40')
```

QSPCENV: Specifies whether users by default are to run in a special environment. The default is to not run in a special environment. This can be overridden in each user profile. If you want all users by default to be running in the System/36 environment, enter the following command:

```
CHGSYSVAL SYSVAL(QSPCENV) VALUE('*S36')
```

Subsystem Configurations Shipped by IBM

Two complete subsystem configurations are supplied by IBM and can be used without changing. The one the system uses when you IPL is controlled by the controlling subsystem description system value (QCTLSBSD) which defaults to 'QBASE QSYS'. This default configuration consists of the following subsystem descriptions:

- QBASE: This is the controlling subsystem. It supports interactive, batch, and communications jobs. It has an autostart job which automatically starts the QSPL subsystem.
- QSPL: This is the spool subsystem. It supports reader and writer jobs.

The other configuration, which is supplied by IBM, consists of the following subsystem descriptions:

- QCTL: This is the controlling subsystem. It only supports signing on at the console. It has an autostart job which automatically starts the QINTER, QBATCH, QCMN, and QSPL subsystems.
- QINTER: This supports all interactive jobs (except at the console).
- QBATCH: This supports all batch jobs.
- QCMN: This supports all communications jobs.

- QSPL: This is the spool subsystem. It supports reader and writer jobs.
- QSNADS: This is the SNADS subsystem. It supports jobs controlling the functions of the SNADS network, and IBM-supplied transaction programs, such as document interchange and object distribution.

The QBASE configuration gives the ability to run all the same functions that you can run with the QCTL configuration and is easier to manage because it consists of fewer subsystems.

The QCTL default configuration allows for more granular control over your system operations by dividing the system activity into different subsystems based on the type of activity. For example, if you want to run batch jobs over the weekend or overnight but do not want anyone to be able to sign on (except at the console), you can easily do that with the QCTL configuration by simply ending the QINTER subsystem. Another reason you might want to end one subsystem without affecting others would be to change some of the subsystem description attributes without an IPL. Many of these cannot be changed while the subsystem is active. If you need to change one of these attributes and are using the QBASE configuration, you would need to make a copy of QBASE with your changes, change the QCTLSBSD system value, and then IPL the system.

If you are considering creating your own subsystem configuration, you may also find that it is easier to use the QCTL configuration as a starting point than the QBASE configuration.

If you want to use the IBM-supplied QCTL subsystem configuration, enter the following command, which will take effect on the next IPL:

```
CHGSYSVAL SYSVAL(QCTLSBSD) VALUE('QCTL QSYS')
```

Appendix D. Work Management APIs

The work management APIs (application programming interfaces) are:

Change Current Job (QWCCCJOB)
Changes information for the current job.

Change Pool Attributes (QUSCHGPA)
Changes the size, activity level, and page option of system storage pools.

Change Pool Tuning Information (QWCCHGTN)
Changes information about tuning being performed on the system for the different storage pools.

Control Trace (QWTCTLTR)
Turns the trace function on and off.

Dump Flight Recorder (QWTDMPFR)
Dumps the contents of the flight recorders for jobs that have them.

Dump Lock Flight Recorder (QWTDMPLF)
Dumps the contents of the lock flight recorder for the device that is specified in the parameter that is passed to the program.

List Active Subsystems (QWCLASBS)
Retrieves a list of active subsystems.

List Job (QUSLJOB)
Lists some or all jobs on the system.

List Job Schedule Entries (QWCLSCDE)
Lists the entries in the job schedule QDFTJOBSCD.

List Subsystem Job Queues (QWDLSJBQ)
Lists the job queues for a subsystem.

Retrieve Data Area (QWCRDTAA)
Retrieves the contents of a data area.

Retrieve Job Description Information (QWDRJOBD)
Retrieves information from a job description object.

Retrieve Job Information (QUSRJOBI)
Retrieves information, such as job attributes and performance data about a specific job.

Retrieve Job Queue Information (QSPRJOBQ)
Retrieves information associated with a specified job queue.

Retrieve Network Attributes (QWCRNETA)
Retrieves network attributes.

Retrieve Subsystem Information (QWDRSBSD)
QWDRSBSD retrieves information about a specific subsystem.

Retrieve System Status (QWCRSSTS)
Retrieves a group of statistics that represent the current status of the system.

Retrieve System Values (QWCRSVAL)
Retrieves system values.

Set Lock Flight Recorder (QWTSETLF)
Runs the lock flight recorder on and off.

Set Trace (QWTSETTR)
Starts the Trace Job (TRCJOB) command for the job passed on the job and user name parameter at the earliest point while the job is starting.

For more detail on work management APIs, see the *System Programmer's Interface Reference*.

Bibliography

The following AS/400 manuals contain information you may need. The manuals are listed with their full title and base order number. When these manuals are referred to in this manual (guide), the short title listed is used.

- *System Operator's Guide*, SC41-8082. This guide provides the system operator or system administrator with information about how to use the system unit operator panel; send and receive messages; respond to error messages; start and stop the system; perform an initial program load (IPL); use the display station function keys; control devices; and also process and manage jobs on the system.

  Short Title: *Operator's Guide.*

Programming Manuals

- *Advanced Backup and Recovery Guide*, SC41-8079. This guide provides the programmer with information about the different media available to save and protect system data as well as a description of how to record changes made to database files and how that information can be used for system recovery and activity report information.

  Short Title: *Advanced Backup and Recovery Guide.*

- *Basic Backup and Recovery Guide*, SC41-0036. This guide contains a subset of the information found in the *Advanced Backup and Recovery Guide*, SC41-8079. The manual contains information about planning a backup and recovery strategy, the different types of media available to save and restore system data, save and restore procedures, and disk recovery procedures. It also describes how to install the system again from backup.

  Short Title: *Basic Backup and Recovery Guide.*

- *Data Management Guide*, SC41-9658. This guide describes the characteristics and programming uses of database files and spooled files. It also provides information about applications with access to input and output file data that is external to the application.

  Short Title: *Data Management Guide*

- *Database Guide*, SC41-9659. This guide provides a detailed discussion of the AS/400 database organization, including information on how to create, describe, and update database files on the system.

  Short Title: *Database Guide*

- *Programming: Control Language Programmer's Guide*, SC41-8077. This guide provides the application programmer or programmer with a wide-ranging discussion of the AS/400 programming topics, including a general discussion of objects and libraries, control language (CL) programming, controlling flow and communicating between programs, working with objects in CL programs, and creating CL programs.

  Short Title: *CL Programmer's Guide.*

- *Programming: Control Language Reference*, SC41-0030. This manual provides the application programmer or programmer with a description of the AS/400 control language (CL). Each command is defined, including its syntax diagram, parameters, default values, and keywords.

  Short Title: *CL Reference.*

- *National Language Support Planning Guide*, GC41-9877. This guide describes the concepts of national language support on the AS/400 system. This support provides a means of operating in a multilingual environment.

  Short Title: *National Language Support Planning Guide*

- *Programming: Performance Tools/400 Guide*, SC41-8084. This guide provides the programmer with information about what performance management is, gives an overview of the tools, and describes how the tools can be used to help manage system performance. The manual gives instructions on how to approach the analysis of system performance and how to do system performance measurement, reporting, capacity planning, and application analysis.

  Short Title: *Performance Tools/400 Guide.*

- *Security Reference*, SC41-8083. This manual provides the programmer (or someone who is assigned the responsibilities of a security officer) with information about general security concepts and planning for security on the system. It also includes information for all users about resource security.

  Short Title: *Security Reference.*

- *System Programmer's Interface Reference*, SC41-8223. This manual provides information on how to create, use, and delete objects that help manage system performance, use spooling, and maintain database files efficiently. This manual also includes information on creating and maintaining the programs for system objects and retrieving OS/400 information by working with objects, database files, jobs, and spooling.

  Short Title: *System Programmer's Interface Reference.*

Communications Manuals

- *Communications: Advanced Program-to-Program Communications Programmer's Guide*, SC41-8189. This manual is intended for the programmer responsible for writing application programs that use advanced program-to-program communications (APPC).

  Short Title: *APPC Programmer's Guide.*

- *Communications and Systems Management Guide (Alerts and Distributed Systems Node Executive)*, SC41-9661. This manual provides the system operator, programmer, or system administrator with information for configuring the AS/400 system to use the remote management support (distributed host command facility), the change management support (distributed systems node executive), and the problem management support (alerts).

  Short Title: *Alerts and DSNX Guide.*

- *Communications: Distribution Services Network Guide*, SC41-9588. This manual provides the system operator or system administrator with information about administering data communications applications on the AS/400 system. This guide may also be useful to a programmer who works with data communications functions on the AS/400 system.

  Short Title: *Distribution Services Network Guide.*

- *Communications: Intersystem Communications Function Programmer's Guide*, SC41-9590. This manual provides the application programmer with the information needed to write application programs that use the AS/400 communications and the OS/400-ICF file. It also contains examples of communications programs and describes return codes.

  Short Title: *ICF Programmer's Guide.*

- *Communications: Remote Work Station Guide*, SC41-0002. This guide provides the system administrator or end user with concepts, examples, and information on preparation and configuration for using the display station pass-through function. This guide also contains information about using 3270 remote attachment, the Distributed Host Command Facility (DHCF) network, and the X.21 short hold mode (SHM) network.

  Short Title: *Remote Work Station Guide.*

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

Y

Customer Satisfaction Feedback

Application System/400
Programming:
Work Management Guide
Version 2

Publication No. SC41-8078-02

Overall, how would you rate this manual?

	Very Satisfied	Satisfied	Dissatis-fied	Very Dissatis-fied
Overall satisfaction				

How satisfied are you that the information in this manual is:

Accurate				
Complete				
Easy to find				
Easy to understand				
Well organized				
Applicable to your tasks				
THANK YOU!				

Please tell us how we can improve this manual:

_____

_____

_____

_____

_____

May we contact you to discuss your responses? __ Yes __ No

Phone: (____) _____ Fax: (____) _____

To return this form:

- Mail it
- Fax it
  United States and Canada: **800+937-3430**
  Other countries: **(+1)+507+253-5192**
- Hand it to your IBM representative.

Note that IBM may use or distribute the responses to this form without obligation.

Name _____ Address _____

Company or Organization _____ _____

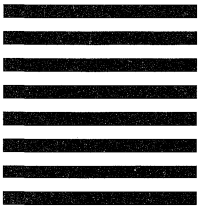
Phone No. _____ _____

Customer Satisfaction Feedback
SC41-8078-02

Cut or Along

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN DEPT 245
IBM CORPORATION
3605 HWY 52 N
ROCHESTER MN 55901-7899

Fold and Tape **Please do not staple** Fold and Tape

Cut or Along

Customer Satisfaction Feedback

Application System/400
Programming:
Work Management Guide
Version 2

Publication No. SC41-8078-02

Overall, how would you rate this manual?

	Very Satisfied	Satisfied	Dissatis-fied	Very Dissatis-fied
Overall satisfaction				

How satisfied are you that the information in this manual is:

Accurate				
Complete				
Easy to find				
Easy to understand				
Well organized				
Applicable to your tasks				
THANK YOU!				

Please tell us how we can improve this manual:

_____

_____

_____

_____

_____

May we contact you to discuss your responses? __ Yes __ No

Phone: (____) _____ Fax: (____) _____

To return this form:

- Mail it
- Fax it
  United States and Canada: **800+937-3430**
  Other countries: **(+1)+507+253-5192**
- Hand it to your IBM representative.

Note that IBM may use or distribute the responses to this form without obligation.

_____ _____
Name Address

_____ _____
Company or Organization

_____ _____
Phone No.

Customer Satisfaction Feedback
SC41-8078-02

IBM®

Cut or
Along

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN DEPT 245
IBM CORPORATION
3605 HWY 52 N
ROCHESTER MN 55901-7899

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or
Along L

Program Number: 5738-SS1

Printed in Denmark by
Scanprint a-s, Viby J.

SC41-8078-02