AS/400e

Client Access Express Host Servers

Version 4

SC41-5740-03

AS/400e

Client Access Express Host Servers

Version 4

SC41-5740-03

Note

Before using this information and the product it supports, be sure to read the information in “Appendix D. Notices” on page 85.

Fourth Edition (May 1999)

This edition replaces SC41-5740-02.

Contents

Figures

Tables

About Client Access Express for Windows (SC41-5740-03)

This information provides brief descriptions of server functions that run on AS/400 and technical information specific to host servers that are used by the Client Access Express product. These may not be all of the servers used by Client Access Express. This book explains how to manage servers and how these servers use AS/400 to provide services for Client Access Express.

Note: If you need information about any other Client Access product's (Client Access for Windows 95/NT or Client Access Enhanced for Windows) host servers, see *Client Access Express for Windows Host Servers*, SC41-5740-03. This book is available from

`http://as400bks.rochester.ibm.com`

This book does not give details about the Client Access Express, but it does explain how the Client Access Express uses the servers. It does not explain how to set up and configure the client environment.

Who should read this book

This book is intended for the system administrator of AS/400 systems operating in a client/server environment. It focuses on the AS/400 server side of the client/server environment.

Before using this book, you should be familiar with the following information:

- AS/400 work management concepts and how your system is set up
- The client environments that your users use (for example Windows 95, Windows 98 or Windows NT based workstations)
- The Client Access Express functions that your users use.

Conventions and terminology used in this book

Throughout this book, PC is used to in place of Personal Computer.

AS/400 Operations Navigator

AS/400 Operations Navigator is a powerful graphical interface for Windows clients. With AS/400 Operations Navigator, you can manage and administer your AS/400 systems from your Windows desktop.

You can use Operations Navigator to manage communications, printing, database, security, and other system operations. Operations Navigator includes Management Central for managing multiple AS/400 systems centrally.

Figure 1 on page x shows an example of the Operations Navigator display:

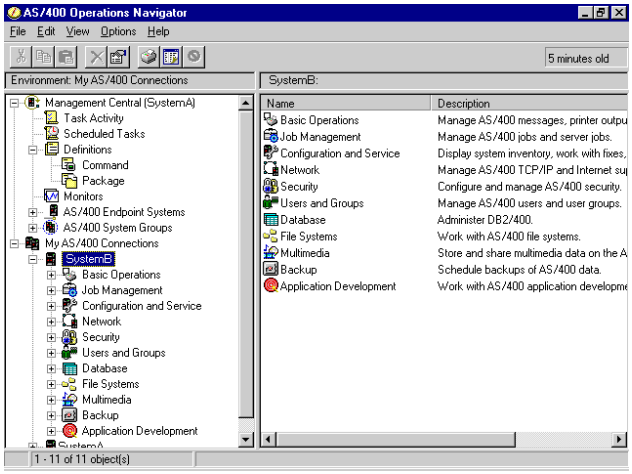


Figure 1. AS/400 Operations Navigator Display

This new interface has been designed to make you more productive and is the only user interface to new, advanced features of OS/400. Therefore, IBM recommends that you use AS/400 Operations Navigator, which has online help to guide you. While this interface is being developed, you may still need to use a traditional emulator such as PC5250 to do some of your tasks.

Installing Operations Navigator subcomponents

AS/400 Operations Navigator is packaged as separately installable subcomponents. If you are installing for the first time and you use the **Typical** installation option, the following options are installed by default:

- Operations Navigator base support
- Basic operations (messages, printer output, and printers)

To install all subcomponents, select the **Full** installation option. To choose the AS/400 Operations Navigator subcomponents you want to install, you can select the **Custom** installation option. (After Operations Navigator has been installed, you can add subcomponents by using Client Access Selective Setup.)

1. Display the list of currently installed subcomponents in the **Component Selection** window of **Custom** installation or Selective Setup.
2. Select AS/400 Operations Navigator.
3. Select any additional subcomponents that you want to install and continue with **Custom** installation or Selective Setup.

Note: To use AS/400 Operations Navigator, you must have Client Access installed on your Windows PC and have an AS/400 connection from that PC. For help in connecting your Windows PC to your AS/400 system, consult *Client Access Express for Windows - Setup*, SC41-5507-00.

After you install Client Access, double-click the **Operations Navigator** icon on your desktop to access Operations Navigator and create an AS/400 connection.

Prerequisite and related information

Use the AS/400 Information Center as your starting point for looking up AS/400 technical information. You can access the Information Center from the AS/400e Information Center CD-ROM (English version: *SK3T-2027*) or from one of these Web sites:

`http://www.as400.ibm.com/infocenter`
`http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm`

The AS/400 Information Center contains important topics such as logical partitioning, clustering, Java, TCP/IP, Web serving, and secured networks. It also contains Internet links to Web sites such as the AS/400 Online Library and the AS/400 Technical Studio. Included in the Information Center is a link that describes at a high level the differences in information between the Information Center and the Online Library.

How to send your comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other AS/400 documentation, fill out the readers' comment form at the back of this book.

- If you prefer to send comments by mail, use the readers' comment form with the address that is printed on the back. If you are mailing a readers' comment form from a country other than the United States, you can give the form to the local IBM branch office or IBM representative for postage-paid mailing.
- If you prefer to send comments by FAX, use either of the following numbers:
  - United States and Canada: 1-800-937-3430
  - Other countries: 1-507-253-5192
- If you prefer to send comments electronically, use one of these e-mail addresses:
  - Comments on books:

    RCHCLERK@us.ibm.com

    IBMMAIL, to IBMMAIL(USIB56RZ)
  - Comments on the AS/400 Information Center:

    RCHINFOC@us.ibm.com

Be sure to include the following:

- The name of the book.
- The publication number of the book.
- The page number or topic to which your comment applies.

Chapter 1. Introduction to Client Access Express Host Servers

This chapter introduces client/server computing and describes how the host servers provided with OS/400 work with Client Access Express to provide client/server functions.

Client/Server Computing

Client/server computing refers to a computing model in which one computer provides services to one or more other computers.

As the term implies, client/server computing has two basic components: a *client* and a *server*. The client requests a service such as running an application, querying a database, printing a document, or even performing a backup or recovery procedure. The server is the resource that handles the client's request. Clients are personal computers and servers are a midrange or mainframe system; however, a server can be another personal computer on the network.

Client/Server Computing with AS/400

AS/400s are full-function servers capable of performing many tasks at once, including file, database, applications, multimedia, mail, print, fax, and wireless communications. When these tasks are handled by several different servers, server management and coordination becomes complex. Having all of your servers on one integrated system greatly reduces the overall cost and complexity of managing your network.

Client Access Express

Client Access Express contains all functions that are needed for connecting PC desktops to AS/400. These functions include the following:

- Communications
- 5250 emulation
- Full network printing capability
- File transfer
- PC file serving
- A full set of application enablers
- Office integration

Client Access Express is a 32-bit client that combines the latest Microsoft Windows technologies with the AS/400 environment so that a single-system view appears on the desktop. Windows 95, 98 or NT screen interfaces, such as My Computer, Windows Explorer, and Network Neighborhood, are typical ways to set up and work with network resources. These interfaces also access AS/400 resources, which provide the PC desktop with a transparent and seamless view of all network resources.

OS/400 Servers

The OS/400 servers discussed in this book are all optimized servers. Most of the OS/400 servers are included in the OS/400 Host Server option (BOSS option 12). You can choose to install them when you install the OS/400 product.

These servers are used by Client Access Express, but are designed so that other client products can also use them. Most of this book focuses on how these servers are used by Client Access Express.

OS/400 Host Server Option

To use Client Access Express, install the Host Server option. Most of the servers discussed in this book are included in the OS/400 Host Server option. File Server is included with the base option of OS/400.

If you are not using any Client Access products and would like to remove the OS/400 Host Server option, you should end the subsystems used by these servers before you remove the option. End the QBASE or QCMN subsystem (for host servers with APPC support), the QSYSWRK and QUSRWRK subsystems (for host servers with sockets support), and the QSERVER subsystem (for database and file server). Problems may occur if you try to delete the option while any of these subsystems are active.

Server to Client Communications

Client Access Express uses TCP/IP to communicate with the AS/400 system servers.

The servers that are provided with the OS/400 Host Server Option use OS/400 sockets support to communicate with clients. The OS/400 sockets support is compatible with Berkeley Software Distributions 4.3 sockets over TCP/IP. Sockets support is provided with the 5769-TC1 product that is installed on the AS/400.

See the following manuals for more information about communications:

- *TCP/IP Configuration and Reference*, SC41-5420-03
- *Sockets Programming*, SC41-5422-03

Servers

The servers include:

- A file server that replaces the shared folder servers
- A database server for Data Transfer, ODBC, Operations Navigator database, SQL APIs (DB APIs) and the Client Access Express OLE DB provider
- A network print server that functions in the same way as the virtual print server but has additional print management functions
- A data queue server
- A remote command and program call server that allows PC applications to issue commands and call programs on the AS/400 and return the results to the client
- A central server that provides services such as license management and other client management functions

- A signon server that provides password management functions for host servers with sockets support
- A server mapper that provides the current server port number to a client requesting a connection

How Client Access Express and the AS/400 Servers Work Together

The following table shows which servers are used with each of the Client Access Express functions. The right-hand column specifies where you can find more information about the servers

Client Access Express for Windows

Table 1. Client Access Express for Windows Functions

Client Function	OS/400 Server Used
Database access APIs • SQL • ODBC APIs	"Database Server" on page 6
Data Transfer	"Database Server" on page 6
ODBC driver	"Database Server" on page 6
Network Drives	"File Server" on page 5
Data queue APIs	"Data Queue Server" on page 7
OLE DB provider	• "Data Queue Server" on page 7 • "Database Server" on page 6 • "Remote Command and Distributed Program Call Server" on page 8
Network print GUI and programming interfaces	"Network Print Server" on page 7
License management Done when an application that requires a license is started (Data Transfer and 5250 emulation)	"Central Server" on page 8
Retrieve conversion map Done only on initial connection if the client does not contain the required conversion maps	"Central Server" on page 8
Remote command functions	"Remote Command and Distributed Program Call Server" on page 8
Distributed program call	"Remote Command and Distributed Program Call Server" on page 8
Send password for validation and change expired password (TCP/IP)	"The Signon Server" on page 9

Chapter 2. Servers

These servers are available with sockets communications support.

File Server

The file server allows clients to store and access information, such as files and programs, located on AS/400. This server replaces the shared folder type 2 server that was used before V3R1. The OS/400 file server interfaces with the integrated file system on AS/400. Its file serving capabilities are equivalent to shared folder, but clients can also access information in any of the new file systems. Clients use their own interface to interact with the file systems, rather than the integrated file system user interfaces and APIs.

The integrated file system is a part of the OS/400 program. It supports stream input/output and storage management similar to personal computer and UNIX operating systems. At the same time, it integrates all information that is stored on the AS/400 system.

The key features of the integrated file system are the following:

- Support for storing information in stream files, which are files that contain long, continuous strings of data. These strings of data might be, for example, the text of a document or the picture elements in a picture. Documents that are stored in AS/400 folders are stream files. Other examples of stream files are PC files and the files in UNIX systems. The stream file support is designed for efficient use in client/server applications.
- A hierarchical directory structure that allows objects to be organized like fruit on the branches of a tree. To access an object, specify the path from the directories to the object.
- A common interface that allows users and applications to access stream files, database files, documents, and other objects that are stored in AS/400.

AS/400 can support several different file systems with similar interfaces. A file system allows users and applications to access specific segments of storage that are organized as logical units. These logical units are files, directories, libraries, and objects. The AS/400 file systems are:

'root'
The '/' file system. The design of this file system takes full advantage of the stream file support and hierarchical directory structure of the integrated file system. It has the characteristics of the DOS and OS/2 file systems.

QFileSvr.400
The OS/400 file server file system. This file system provides transparent access to the Integrated File System (IFS) of remote AS/400 systems.

Note: With the QFileSvr.400 file system, only one job services multiple users.

QOpenSys
The open systems file system. The design of this file system is compatible with UNIX-based system standards, such as POSIX and XPG.

QOPT
The optical support file system. This system provides access to the CD-ROM device and optical media library devices directly attached to AS/400.

QSYS.LIB
The library file system. This file system supports the AS/400 library system. It provides access to database files and all other AS/400 object types managed by the library support.

QDLS
The document library services file system. This file system supports the folders structure. It provides access to documents and folders.

QLANSrv
The LAN Server/400 file system. This file system provides access to the same directories and files that are accessed through the LAN Server/400 licensed program.

For more information on the integrated file system, see *Integrated File System Introduction*, SC41-5711-02.

The OS/400 file server can give clients access to all of the AS/400 file systems or just QDLS, depending on the support provided by the client product.

The programs listed in the following table are included with this server.

Table 2. File Server objects

Program name	Library	Object type	Description
QPWFSERVSO	QSYS	*PGM	Server program
QPWFSERVS2	QSYS	*PGM	Server program
QPWFSERVSD	QSYS	*PGM	Daemon program
QPWFSERV	QSYS	*JOBD	Job description used for server jobs
QPWFSERVER	QSYS	*CLS	Class used for all file server and database server jobs
QPWFSERVSS	QSYS	*PGM	SSL server program

Database Server

The database server allows clients access to the functions included with DB2/400. This server provides:

- Support for remote SQL access
- Access to data through ODBC interfaces
- Database functions (such as creating and deleting files and adding and removing file members)
- Retrieval functions for obtaining information about database files that exist on the system (such as SQL catalog functions)

The programs listed in the following table are included with this server.

Table 3. Database Server programs

Program name	Library	Description
QZDASOINIT	QIWS	Server program
QZDASON2	QIWS	Sockets setup program
QZDASRVSD	QIWS	Daemon program

Table 3. Database Server programs (continued)

Program name	Library	Description
QZDASSINIT	QIWS	SSL server program
Note: The *PGM objects QZDANDB, QZDAROI, QZDASQL, and QZDACMDP are used by the database server.		

Data Queue Server

The optimized data queue server provides the same function as the original data queue server, but with several enhancements. The data queue server was also changed to support the common data stream for all the optimized servers and clients.

The programs listed in the following table are included with this server.

Table 4. Data Queue server program provided for use with sockets support

Program name	Library	Description
QZHQSSRV	QIWS	Server program
QZHQSRVD	QIWS	Daemon program

Network Print Server

The OS/400 network print server allows enhanced client control over print resources on AS/400. This print server provides the following capabilities to each client by requesting print serving:

Spooled file
Create, seek, open, read, write, close, hold, release, delete, move, send, call exit program, change attributes, retrieve message, answer message, retrieve attributes, and list

Writer job
Start, end, and list

Printer device
Retrieve attributes and list

Output queue
Hold, release, purge, list, and retrieve attributes

Library
List

Printer file
Retrieve attributes, change attributes, and list

Network print server
Change attributes and retrieve attributes

The programs listed in the following table are included with this server.

Table 5. Network Print server

Program name	Library	Description
QNPSERVS	QIWS	Server program
QNPSERVD	QIWS	Daemon program

Central Server

The central server provides the following services for clients:

- License management

  The initial request from either Data Transfer or PC5250 reserves a license for that Client Access Express user. The server remains active until the release delay timeout expires. This delay time default is two hours but can be customized. Use Operation Navigator to view the AS/400 system's properties. This shows if any licenses are being used.

- Retrieve conversion map

  The central server retrieves conversion maps for clients who need them. These conversion maps are usually used for ASCII to EBCDIC conversions and for EBCDIC to ASCII conversions. The client can request a map by giving the correct source, the target coded character set identifier (CCSID), and a table of code points to be converted. The server then returns the correct mapping for the client to use.

The programs listed in the following table are included with this server.

Table 6. Central server programs

Program name	Library	Description
QZSCSRVS	QIWS	Server program
QZSCSRVSD	QIWS	Daemon program

Remote Command and Distributed Program Call Server

The remote command and distributed program call server allows client users and applications to issue AS/400 CL commands and call programs.

The remote command allows the user to run multiple commands in the same job. It also offers a better security check for AS/400 users with limited capabilities (LMTCPB =*YES) in their user profile.

The distributed program call support allows client applications to call AS/400 programs and pass parameters (input and output). After the program runs on AS/400, the output parameter values return to the client application. This process allows applications to access AS/400 resources easily without worry about the communications and conversions that must take place.

The programs listed in the following table are included with this server.

Table 7. Remote Command and Distributed Program Call server programs

Program name	Library	Description
QZRCSRVS	QIWS	Server program
QZRCSRVSD	QIWS	Daemon program

The Signon Server

The signon server provides security for clients that use TCP/IP communications support. This security function allows clients to prevent access to the system for users with expired passwords.

The programs listed in the following table are included with this server.

Table 8. Signon server programs

Program name	Library	Description
QZSOSIGN	QIWS	Server program
QZSOSGND	QIWS	Daemon program

The Server Port Mapper

The port mapper provides a way for the client to find the port for a particular service (server).

The program listed in the following table is included with this server.

Table 9. Server port mapper

Program name	Library	Description
QZSOSMAPD	QIWS	Server port mapper program

Chapter 3. AS/400 system configuration

When you install OS/400 on your AS/400, several system-wide attributes are set at initial values. These attributes are called system values. Values are also set to help control your communications environment. These values are called network attributes.

To meet your system's requirements, you need to understand these default values and how to change them. Some of these values are of particular interest if you are running in a client/server environment.

To understand how to manage your AS/400 system, you should review the book *Work Management*, SC41-5306-03, before you continue with this book.

This chapter describes some of the system values that the OS/400 servers and Client Access Express use and how they affect your system.

System Values on the AS/400

A system value contains control information that operates certain parts of the system. A user can change the system values to define the work environment. Examples of system values are system date and library list.

The AS/400 has many system values; the following values are of particular interest in a client/server environment.

QAUDCTL
Audit control. This system value contains the on and off switches for object and user level auditing. Changes that are made to this system value take effect immediately.

QAUDENDACN
Audit journal error action. This system value specifies the action the system takes if errors occur when an audit journal entry is being sent by the operating system security audit journal. Changes that are made to this system value take effect immediately.

QAUDFRCLVL
Force audit journal. This system value specifies the number of audit journal entries that can be written to the security auditing journal before the journal entry data is forced to auxiliary storage. Changes that are made to this system value take effect immediately.

QAUDLVL
Security auditing level. Changes made to this system value take effect immediately for all jobs running on the system.

QAUTOVRT
Determines whether the system should automatically create virtual devices. This is used with display station pass-through and Telnet sessions.

QCCSID
The coded character set identifier, which identifies:

- 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 needed by the system

This value is based on the language that is installed on the system. It determines whether data must be converted to a different format before being presented to the user. The default value is 65535, which means this data is not converted.

QCTLSBSD
The controlling subsystem description

QDSPSGNINF
Determines whether the signon information display shows after signon by using the 5250 emulation functions (workstation function, PC5250).

QLANGID
The default language identifier for the system. It determines the default CCSID for a user's job if the job CCSID is 65535. The clients and servers use this default job CCSID value to determine the correct conversion for data that is exchanged between the client and the server.

QLMTSECOFR
Controls whether a user with all-object (*ALLOBJ) or service (*SERVICE) special authority can use any device. If this value is set to 1, all users with *ALLOBJ or *SERVICE special authorities must have specific *CHANGE authority to use the device.

This affects virtual devices for 5250 emulation. The shipped value for this is 1. If you want authorized users to sign on to PCs, you must either give them specific authority to the device and controller that the PC uses or change this value to 0.

QMAXSIGN
Controls the number of consecutive incorrect signon attempts by local and remote users. Once the QMAXSIGN value is reached, the system determines the action with the QMAXSGNACN system value.

If the QMAXSGNACN value is 1 (vary off device), the QMAXSIGN value does not affect a user who enters an incorrect password on the PC when they are starting the connection.

This is a potential security exposure for PC users. The QMAXSGNACN should be set to either 2 or 3.

QMAXSGNACN
Determines what the system does when the maximum number of signon attempts is reached at any device. You can specify 1 (vary off device), 2 (disable the user profile) or 3 (vary off device and disable the user profile). The shipped value is 3.

QPWDEXPITV
The number of days for which a password is valid. Changes that are made to this system value take effect immediately.

QPWDLMTAJC
Limits the use of adjacent numbers in a password. Changes that are made to this system value take effect the next time a password is changed.

QPWDLMTCHR
Limits the use of certain characters in a password. Changes that are made to this system value take effect the next time a password is changed.

QPWDLMTREP
Limits the use of repeating characters in a password. Changes that are made to this system value take effect the next time a password is changed.

QPWDMAXLEN
The maximum number of characters in a password. Changes that are made to this system value take effect the next time a password is changed.

QPWDMINLEN
The minimum number of characters in a password. Changes that are made to this system value take effect the next time a password is changed.

QPWDPOSDIF
Controls the position of characters in a new password. Changes that are made to this system value take effect the next time a password is changed.

QPWDRQDDGT
Requires a number in a new password. Changes that are made to this system value take effect the next time a password is changed.

QPWDRQDDIF
Controls whether the password must be different than previous passwords.

QPWDVLDPGM
Password validation program name and library that are supplied by the computer system. Both an object name and library name can be specified. Changes that are made to this system value take effect the next time a password is changed.

QRMTSIGN
Specifies how the system handles remote signon requests. A TELNET session is actually a remote signon request. This value determines several actions, as follows:

- '*FRCSIGNON': All remote sign-on sessions are required to go through normal sign-on processing.
- '*SAMEPRF': For 5250 display station pass-through or workstation function, when the source and target user profile names are the same, the sign-on may be bypassed for remote sign-on attempts. When using TELNET, the sign-on may be bypassed.
- '*VERIFY': After verifying that the user has access to the system, the system allows the user to bypass the sign-on.
- '*REJECT': Allows no remote sign-on for 5250 display station pass-through or work station function. When QRMTSIGN is set to *REJECT, the user can still sign on to the system by using TELNET. These sessions will go through normal processing. If you want to reject all TELNET requests to the system, end the TELNET servers.
- '*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. This option is only valid for passthrough.

This value also specifies a program name to run that determines which remote sessions are to be allowed.

The shipped value is *FRCSIGNON. If you want users to be able to use the bypass signon function of the 5250 emulator, change this value to *VERIFY.

QSECURITY

System security level. Changes that are made to this system value take effect at the next IPL.

- 20 means that the system requires a password to signon.
- 30 means that the system requires password security at signon and object security at each access. You must have authority to access all system resources.
- 40 means that the system requires password security at signon and object security at each access. Programs that try to access objects through unsupported interfaces fail.
- 50 means that the system requires password security at signon, and users must have authority to access objects and system resources. The security and integrity of the QTEMP library and user domain objects are enforced. Programs that try to access objects through interfaces that are not supported or that try to pass unsupported parameter values to supported interfaces will fail.

QSTRUPPGM

The program that runs when the controlling subsystem starts or when the system starts. This program performs setup functions such as starting subsystems.

QSYSLIBL

The system part of the library list. This part of the library list is searched before any other part. Some client functions use this list to search for objects.

Chapter 4. Managing AS/400 servers

This chapter describes how to manage the OS/400 server jobs. It describes the subsystems in which the servers run, the objects that affect the servers, and how to manage these resources.

The servers shipped with the OS/400 program do not typically require any changes to your existing system configuration in order to work correctly. They are set up and configured when you install OS/400. You may want to change the way the system manages the server jobs to meet your needs, solve problems, improve system performance, or simply view the jobs on the system. To make such changes and meet processing requirements, you must know which objects affect which pieces of the system and how to change those objects. To really understand how to manage your AS/400 system, carefully review the book *Work Management*, SC41-5306-03, before you continue with this chapter.

Sockets Communication Support

Several concepts pertain specifically to the sockets communications support that are used by the optimized servers. Figure 2 describes these concepts in detail.

Establishing Client/Server Communications

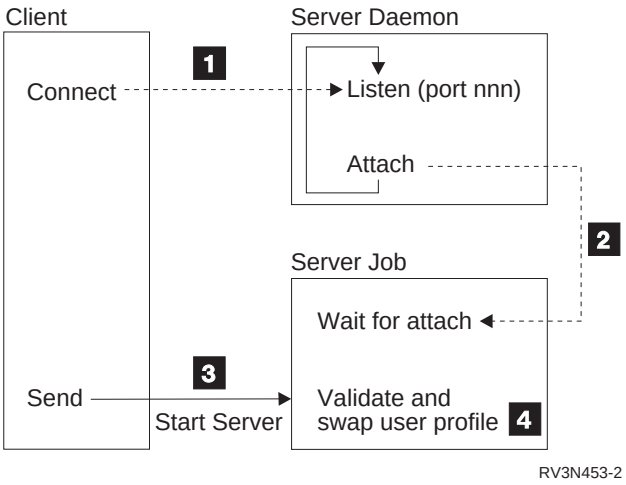


Figure 2. Establishing client/server communications

To initiate a server job that uses sockets communications support, the client system connects to a particular server's port number **1** . A server daemon must be started (with the STRHOSTSVR command) to listen for and accept the client's connection request. Upon accepting the connection request, the server daemon issues an internal request to attach the client's connection to a server job **2** . This server job may be a prestarted job or, if prestart jobs are not used, a batch job that is submitted when the client connection request is processed. The server job handles any further communications with the client.

The initial data exchange includes a request that identifies the user profile and password that are associated with the client user **3** . Once the user profile and

password are validated, the server job switches to this user profile and changes the job by using many of the attributes defined for the user profile, such as accounting code and output queue 4 .

Service Table

As described in “Establishing Client/Server Communications” on page 15, each type of server has its own server daemon, which listens on a port for incoming client connection requests. There are exceptions to this. For instance, the transfer function over sockets uses the database server daemon; the network drive server uses the file server daemon; and the virtual print server uses the network print server daemon. In addition, the server mapper daemon also listens on a specified port, and allows a client to obtain the current port number for a specified server. Table 10 provides the port numbers for each of the server daemons and the server mapper daemon and their symbolic service names.

Host Servers Service Table Entries

The following table shows the initial service table entries provided for the optimized servers and server mapper that use sockets over TCP communication support.

Table 10. Port numbers for host servers and server mapper

Service name	Description	Port number
as-central	Central server	8470
as-database	Database server	8471
as-dtaq	Data queue server	8472
as-file	File server	8473
as-netprt	Network print server	8474
as-rmtcmd	Remote command/program call server	8475
as-signon	Signon server	8476
as-svrmap	Server mapper	449

The following table shows the initial service table entries provided for the servers and server mapper that use SSL (Secure Sockets Layer) support.

Table 11. Port numbers for host servers and server daemons

Service name	Description	Port number
as-central-s	Secure Central server	9470
as-database-s	Secure Database server	9471
as-dtaq-s	Secure Data queue server	9472
as-file-s	Secure File server	9473
as-netprt-s	Secure Network print server	9474
as-rmtcmd-s	Secure Remote command/program call server	9475
as-signon-s	Secure Signon server	9476

Each of the server daemons listen on the port number that is provided in the service table for the specified service name. For example, the network print server daemon, with the initial configuration that is provided, listens on port number 8474, which is associated with service name ’as-netprt.’

The port numbers for each server daemon are not fixed; the service table can be modified by using different port numbers if your installation requires such changes. However, the service name must remain the same as that shown in Table 10 on page 16 or Table 11 on page 16. Otherwise, the server daemons cannot establish a port number on which to accept incoming requests for client connection.

If a new service table entry is added to identify a different port number for a service, any pre-existing service table entries for that service name should be removed. Removing these entries eliminates the duplication of the service name in the table and eliminates the possibility of unpredictable results when the server daemon starts.

The server mapper daemon listens on the well-known port for TCP/IP. The well-known port number for TCP/IP is 449. The well-known port numbers is reserved for the exclusive use of the OS/400 Host Servers. Therefore, the entry for the 'as-svrmap' service name should **not** be removed from the service table.

Displaying and Modifying Service Table Entries

You can use the WRKSRVTBLE command to display the service names and their associated port numbers.

```
                  Work with Service Table Entries
                                                          System:   AS400597
Type options, press Enter.
  1=Add   4=Remove   5=Display

Opt  Service                                  Port  Protocol
 _   ___________________________________  _________  ____________________
 _   as-central                             8470  tcp
 _   as-database                            8471  tcp
 _   as-dtaq                                8472  tcp
 _   as-file                                8473  tcp
 _   as-netprt                              8474  tcp
 _   as-rmtcmd                             8475  tcp
 _   as-signon                              8476  tcp
 _   as-svrmap                              449  tcp
        .
        .
        .
```

By selecting option 5 (display) for any entry, you also see the alias names. Use the ADDSRVTBLE and RMVSRVTBLE commands to change the service table for your installation.

Server Mapper Daemon

The server mapper daemon is a batch job that runs in the QSYSWRK subsystem. It provides a method for client applications to determine the port number associated with a particular server.

This job listens on a well-known port for a connection request from a client. The well-known port number for TCP/IP is 449. The client sends the service name to the server mapper. The server mapper obtains the port number for the specified service name from the service table. The server mapper returns this port number to the client, ends the connection, and returns to listen for another connection request. The client uses the port number returned from the server mapper daemon to connect to the specified server daemon.

The server mapper daemon starts with the STRHOSTSVR command and ends with the ENDHOSTSVR command.

Server Daemons

The server daemon is a batch job associated with a particular server type. There is only one server daemon for each of the different server types (such as database, network print, and signon). Each server type has a one-to-many relationship between its server daemon and the actual server jobs; one server daemon potentially has many associated server jobs.

The server daemon allows client applications to start communications with a host server that is using sockets communications support. The server daemon does this by handling and routing incoming connection requests. Once the client establishes communications with the server job, there is no further association between the client and the server daemon for the duration of that server job.

All of the server daemons, except the database and file server daemons, run in the QSYSWRK subsystem. The database and file server daemons run in the QSERVER subsystem.

The server daemons are started with the STRHOSTSVR command. The server daemons must be active in order for client applications to establish a connection with a host server that is using sockets communications support.

If you are starting the database or file server daemons, the QSERVER subsystem must be active. If you start any of the other server daemons, the QSYSWRK subsystem must be active.

Start Host Server (STRHOSTSVR) Command

The Start Host Server (STRHOSTSVR) command starts the host server daemons and the server mapper daemon.

In addition, the STRHOSTSVR command attempts to start the prestart jobs that are associated with the specified server types. There is no prestart job entry that is associated with the server mapper (*SVRMAP) type.

One server daemon exists for each of the host server types. In addition, one server mapper daemon is associated with all host servers which supports client applications seeking to obtain a particular port number for a host server daemon. The client application uses this port number then to connect to the host server daemon. The server daemon accepts the incoming connection request and routes it to the server job for further processing.

The daemons are batch jobs that are submitted to either the QSYSWRK or the QSERVER subsystem. This depends on the value or values that are specified for the SERVER keyword. All daemon jobs are submitted to the QSYSWRK subsystem, with the exception of the *DATABASE and *FILE server daemons, which are submitted to the QSERVER subsystem. There are no server jobs that are associated with the server mapper daemon.

In order for the server daemons and the server mapper daemon to start successfully, the QSYSWRK subsystem and, for *DATABASE and *FILE server, the QSERVER subsystem must be active. If the required subsystem is not active, then

the submission of the daemon job fails. QUSRWRK or the user-defined subsystem needs to be active for the server jobs to run.

To have the host servers start automatically when TCP/IP is started:

1. Double-click **Operations Navigator —> Network —> Servers —> Client Access**.
2. Right-click the server that you want to start and select **Properties**.
3. Put a check in the box **Start when TCP/IP is started**. Now the jobs for that server will start when TCP/IP is started.

Client Access servers support TCP/IP connections that use the Secure Sockets Layer (SSL). SSL encrypts all data that is sent between the client and server. To use SSL, you must have a certificate assigned to the host server. Certificates are assigned using digital certificate manager. For more information, go to this url.

`http://publib.boulder.ibm.com/html/as400/infocenter.html`

Go to **Client Access**, **Administration**, **Secure Sockets administration**.

See *Client Access Express for Windows - Setup*, SC41-5507-00 for information on how to establish SSL communications from the Client Access Express for Windows client.

The RQDPCL keyword is used to identify which communication protocols (TCP/IP or IPX) must be active at the time the STRHOSTSVR command is issued. If the required protocols are not active, the STRHOSTSVR command will fail. The server daemon can be started without any active protocols. When used in conjunction with the QZBSEVTM autostart job, the daemon job can dynamically identify when protocols become active.

Restrictions

- This command enables client applications to communicate with any of the host servers that use sockets communications support.
- Only one server daemon can be active for a specific server type. Requests to start a server daemon that is already active issues a message to the command issuer.

Note: The host servers will not start if the QUSER password has expired. The password expiration interval should be set to *NOMAX for the QUSER profile. With this value, the password will not expire.

Parameters

The following parameters specify the host server daemons to be started by this command.

SERVER

***ALL**
All of the host server daemons and the server mapper daemon are started. If specified, no other special values are accepted.

***CENTRAL**
The central server daemon is started in the QSYSWRK subsystem. The daemon job is QZSCSRVSD. The associated server job is QZSCSRVS.

***DATABASE**
The database server daemon is started in the QSERVER subsystem. The daemon job is QZDASRVSD. The associated server jobs are QZDASOINIT, QZDASSINIT and QTFPJTCP.

***DTAQ**
The data queue server daemon is started in the QSYSWRK subsystem. The daemon job is QZHQSRVD. The associated server job is QZHQSSRV.

***FILE**
The file server daemon is started in the QSERVER subsystem. The daemon job is QPWFSERVSD. The associated server jobs are QPWFSERVSO, QPWFSERVSS, and QPWSERVS2

***NETPRT**
The network print server daemon is started in the QSYSWRK subsystem. The daemon job is QNPSERVD. The associated server jobs are QNPSERVS and QIWVPPJT.

***RMTCMD**
The remote command and distributed program call server daemon is started in the QSYSWRK subsystem. The daemon job is QZRCSRVSD. The associated server job is QZRCSRVS.

***SIGNON**
The signon server daemon is started in the QSYSWRK subsystem. The daemon job is QZSOSGND. The associated server job is QZSOSIGN.

***SVRMAP**
The server mapper daemon is started in the QSYSWRK subsystem. The daemon job is QZSOSMAPD.

Optional parameter

RQDPCL

Single values

***ANY**
At least one of the possible communication protocols must be active at the time the STRHOSTSVR command is issued. If no protocols are active, escape message PSW300D will be issued and the host server daemons will not be started. A diagnostic message (PWS3008 or PWS300F) will be issued for each protocol that is found to be inactive.

***NONE**
No communication protocol must be active at the time the STRHOSTSVR command is issued for the host server daemons to start. No messages will be issued for protocols which are inactive.

Specific protocol values

***TCP**
The TCP/IP communication protocol must be active at the time the STRHOSTSVR command is issued. If TCP/IP is not active, diagnostic message PWS3008 and escape message PWS300D will be issued and the host server daemons will not be started.

***IPX**
The IPX communication protocol must be active at the time the STRHOSTSVR

command is issued. If IPX is not active, diagnostic message PWS3008 and escape message PWS300D will be issued and the host server daemons will not be started.

Examples

Example 1: Starting all host server daemons

```
STRHOSTSVR SERVER(*ALL)
```

This command starts all of the server daemons and the server mapper daemon, as long as at least one communication protocol is active.

Example 2: Starting specific server daemons

```
STRHOSTSVR SERVER(*CENTRAL *SVRMAP)
           RQDPCL(*NONE)
```

This command starts the central server daemon and the server mapper daemon in the QSYSWRK subsystem, even if no communication protocols are active.

Example 3: Specifying one required protocol

```
STRHOSTSVR SERVER(*ALL) RQDPCL(*TCP)
```

This command starts all of the host server daemons and the server mapper daemon, as long as the TCP/IP communication protocol is active.

STRHOSTSVR Command Prompt

```
                      Start Host Server Daemon (STRHOSTSVR)

Type choices, press Enter.

Server type  . . . . . .  . .   _____       *ALL, *CENTRAL, DATABASE...
               + for more values _____
Required Protocol. . . . . . .  *ANY_       *ANY, *NONE, *TCP, *IPX
                                _____

                                                                    Bottom
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

Figure 3. STRHOSTSVR command prompt

If a '?' value is entered on the first line under **Server type**, the following display appears:

```
                  Specify Value for Parameter SERVER

Type choice, press Enter.


Server type  . . . . . .

Single Values
  *ALL
Other Values
  *CENTRAL
  *DATABASE
  *DTAQ
  *FILE
  *NETPRT
  *RMTCMD
  *SIGNON
  *SVRMAP




F3=Exit   F5=Refresh   F12=Cancel   F13=How to use this display   F24=More keys
```

Figure 4. STRHOSTSVR command prompt values

If a '+' value is entered on the second line under **Server type**, the following display appears:

```
                  Specify More Values for Parameter SERVER

Type choices, press Enter.

Server type  . . . . . .          *ALL, *CENTRAL, *DATABASE...

                                                                      Bottom
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

Figure 5. STRHOSTSVR command SERVER prompt

If a '?' value is entered on the first line under **Required Protocol**, the following display appears:

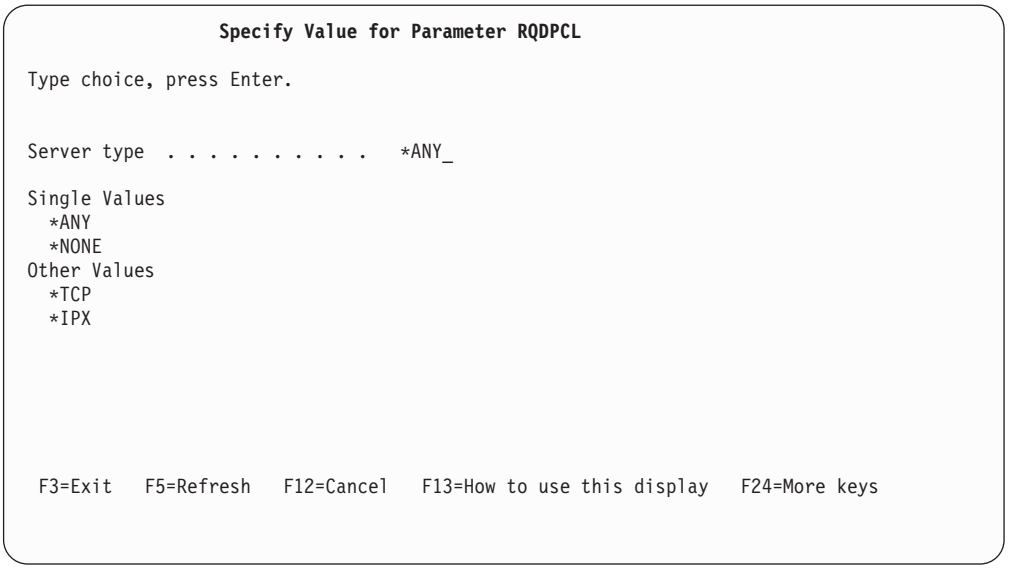


Figure 6. STRHOSTSVR command RQDPCL prompt

If a '?' value is entered on the second line under **Required Protocol**, the following display appears:

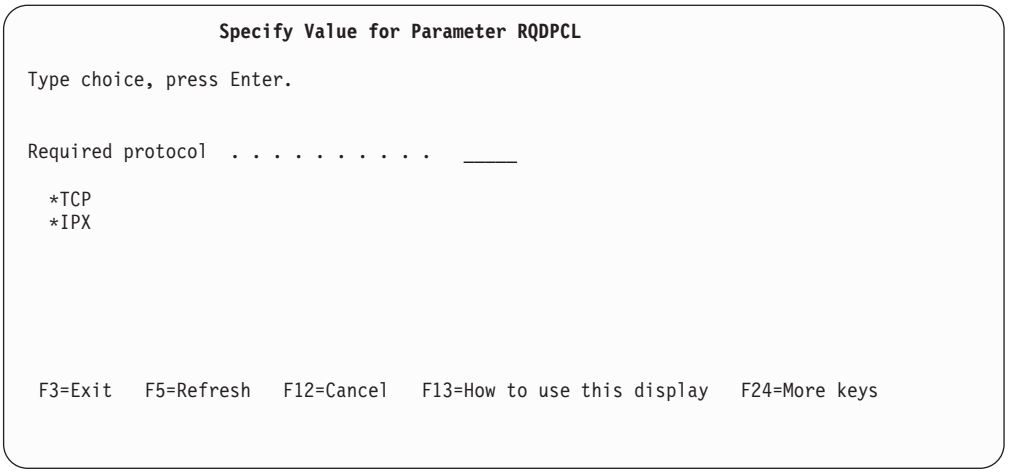


Figure 7. STRHOSTSVR command RQDPCL prompt

End Host Server (ENDHOSTSVR) Command

The End Host Server (ENDHOSTSVR) command ends the host server daemons and the server mapper daemon.

If a server daemon is ended, while servers of that type are connected to client applications, the server jobs remain active until communication with the client application ends. Subsequent connection requests from the client application to that server will fail until the server daemon is started again.

If the server mapper daemon ends, any existing client connections to the server jobs are unaffected. Subsequent requests from a client application to connect to the server mapper (to obtain a server daemon port number) will fail until the server mapper is started again.

Restrictions

- If the End Host Server command ends a particular daemon that is not active, a diagnostic message is issued. This same diagnostic message is not sent for any daemons that are not active when an ENDHOSTSVR SERVER(*ALL) command is issued. A request to end *ALL host server daemons ends any active daemons.

Parameters

The following parameters specify the host server daemons to end by using this command.

***ALL**
If active, server daemons and the server mapper daemon end. If specified, no other special values are accepted.

***CENTRAL**
The central server daemon in the QSYSWRK subsystem ends.

***DATABASE**
The database server daemon in the QSERVER subsystem ends.

***DTAQ**
The data queue server daemon in the QSYSWRK subsystem ends.

***FILE**
The file server daemon in the QSERVER subsystem ends.

***NETPRT**
The network print server daemon in the QSYSWRK subsystem ends.

***RMTCMD**
The remote command and distributed program call server daemon in the QSYSWRK subsystem ends.

***SIGNON**
The signon server daemon in the QSYSWRK subsystem ends.

***SVRMAP**
The server mapper daemon in the QSYSWRK subsystem ends.

Examples

Example 1: Ending all host server daemons

```
ENDHOSTSVR SERVER(*ALL)
```

This command ends all active server daemons and the server mapper daemon.

Example 2: Ending specific server daemons

```
ENDHOSTSVR SERVER(*CENTRAL *SVRMAP)
```

This command ends the central server daemon and the server mapper daemon.

ENDHOSTSVR Command Prompt

```
                        End Host Server Daemon (ENDHOSTSVR)

Type choices, press Enter.

Server type  . . . . . .          *ALL, *CENTRAL, DATABASE...
               + for more values

                                                                      Bottom
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

Figure 8. ENDHOSTSVR command prompt

If a '?' value is entered on the first line, then the following display appears:

```
                  Specify Value for Parameter SERVER

Type choice, press Enter.


Server type  . . . . . .

Single Values
  *ALL
Other Values
  *CENTRAL
  *DATABASE
  *DTAQ
  *FILE
  *NETPRT
  *RMTCMD
  *SIGNON
  *SVRMAP



F3=Exit   F5=Refresh   F12=Cancel   F13=How to use this display   F24=More keys
```

Figure 9. ENDHOSTSVR command prompt values

If a '+' value is entered on the second line, then the following display appears:

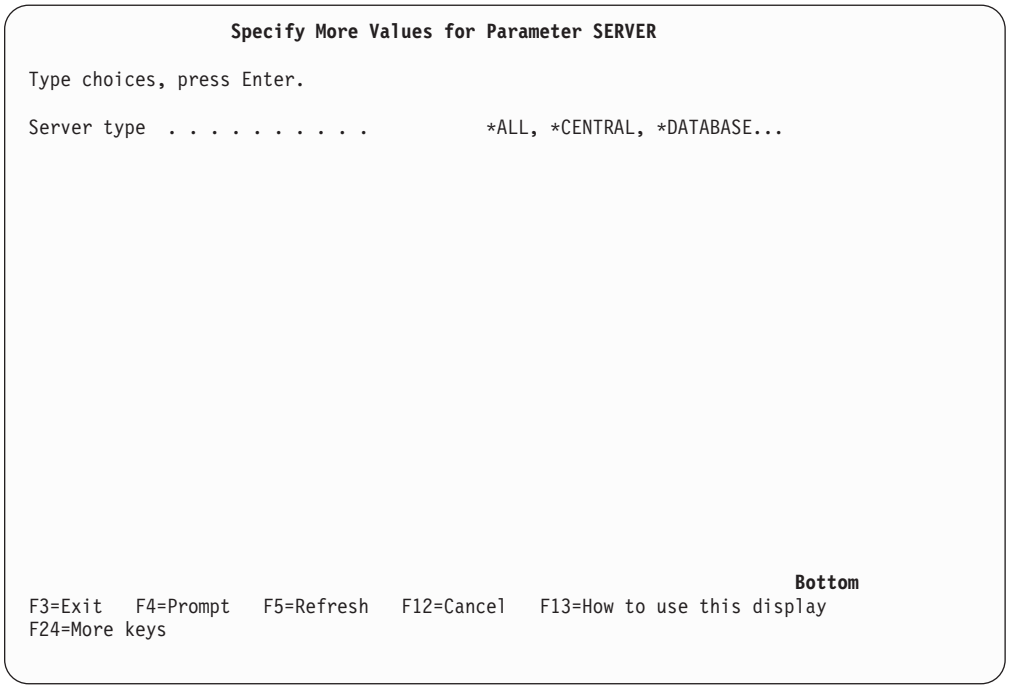


Figure 10. ENDHOSTSVR command SERVER prompt

Subsystems on the AS/400

The following sections describe which system-supplied subsystems are used for each of the server functions. These sections also detail how the subsystem descriptions relate to the server jobs.

Subsystems Used for Server Jobs

The server jobs are configured to run in different subsystems, depending on their function. The following are the subsystems used for the server jobs.

QSYSWRK

All of the daemon jobs (with the exception of the database and file servers, and their associated daemon jobs) run in this subsystem. The file server and database server run in the QSERVER subsystem.

QUSRWRK

This subsystem is where the server jobs run for these servers:

- Network Print
- Remote Command/Program Call
- Central
- Data Queue
- Signon

QSERVER

The file server, and the database server and their associated daemon jobs must run in this subsystem.

If this subsystem is not active, requests to establish a connection to the file server or the database server daemons fail.

Automatically Starting Subsystems

The QSYSWRK subsystem starts automatically when you IPL, regardless of the value specified for the controlling subsystem.

If you use the default startup program provided with the system, the QSERVER and QUSRWRK subsystems start automatically when you IPL. The system startup program is defined in the QSTRUPPGM system value, and the default value is QSTRUP QSYS.

If you want to change the system startup, you can change the QSTRUPPGM system value to call your own program. You can use the shipped program QSTRUP in QSYS as a base for the start-up program that you create.

Note: If you use the database server or file server and you made changes to the system startup, you must ensure that the QSERVER subsystem is added to the list of subsystems that are started in the program specified by the QSTRUPPGM system value.

Subsystem Descriptions

A subsystem description defines how, where, and how much work enters a subsystem, and which resources the subsystem uses to do the work. The following sections describe which parts of the subsystem descriptions affect the servers.

Autostart Jobs

Autostart jobs perform one-time initialization or do repetitive work that is associated with a particular subsystem. The autostart jobs associated with a particular subsystem are automatically started each time the subsystem is started.

Prestart Job Entries for Server Jobs

A prestart job is a batch job that starts running before a program on a remote system initiates communications with the server. Prestart jobs use prestart job entries in the subsystem description to determine which program, class, and storage pool to use when the jobs are started. Within a prestart job entry, you must specify attributes for the subsystem to use to create and to manage a pool of prestart jobs.

Prestart jobs increase performance when you initiate a connection to a server. Prestart job entries are defined within a subsystem. Prestart jobs become active when that subsystem is started, or they can be controlled with the Start Prestart Job (STRPJ) and End Prestart Job (ENDPJ) commands.

Use of Autostart Jobs

The QSERVER subsystem has an autostart job defined for the file server and database server jobs. If this job is not running, the servers cannot start, and the QSERVER subsystem ends. If a problem occurs with this job, you may want to end and restart the QSERVER subsystem.

The QSYSWRK subsystem has an autostart job defined for all of the optimized servers. This job monitors for events sent when a STRTCP or STRIPX command has been issued. In this way, the server daemon jobs can dynamically determine when a communication protocol has become active. The daemon jobs will then begin to listen on the appropriate ports. If the autostart job is not active, and a protocol is started, the following sequence of commands must be issued in order to start using the protocol:

1. ENDHOSTSVR *ALL
2. STRHOSTSVR *ALL

The autostart job is named QZBSEVTM. If the job is not active, it can be started by issuing the following command:

```
QSYS/SBMJOB CMD(QSYS/CALL PGM(QSYS/QZBSEVTM)) JOB(QZBSEVTM) JOBD(QSYS/QZBSEJBD)
PRTDEV(*USRPRF) OUTQ(*USRPRF) USER(QUSER) PRTTXT(*SYSVAL) SYSLIBL(*SYSVAL)
CURLIB(*CRTDFT) INLLIBL(*JOBD) SRTSEQ (*SYSVAL) LANGID(*SYSVAL) CNTRYID(*SYSVAL)
CCSID(*SYSVAL)
```

Note: Only one instance of program QZBSEVTM can be running at any one time.

Use of Prestart Jobs

System information that pertains to prestart jobs (such as DSPACTPJ) uses the term 'program start request' exclusively to indicate requests made to start prestart jobs, even though the information may pertain to a prestart job that was started as a result of a sockets connection request.

Notes:

1. Prestart jobs can be reused, but there is no automatic cleanup for the prestart job once it has been used and subsequently returned to the pool. The number of times the prestart job is reused is determined by the value specified for the maximum number of uses (MAXUSE) value of the ADDPJE or CHGPJE CL commands. This means that resources that are used by one user of the prestart job must be cleaned up before ending use of the prestart job. Otherwise, these resources will maintain the same status for the next user that uses the prestart job. For example, a file that is opened but never closed by one user of a prestart job remains open and available to the following user of the same prestart job.
2. By default, some of the server jobs run in QUSRWRK. QUSRWRK has prestart job entries added to it when the host servers option is installed. Using Operations Navigator, you can configure some or all of these servers to run in a subsystem other than QUSRWRK.
   a. Double-click **Operations Navigator —> Network —> Servers —> Client Access**.
   b. Right-click the server that you want to configure subsystems for and select **Server Jobs**.
   c. Right-click on the job that you want to change. Options that are available show in the drop down box.

If you move jobs from QUSRWRK to your own subsystem, you must:

a. create your own subsystem description
b. add your own pre-start job using the ADDPJE command. Set the STRJOBS parameter to *YES.

If you do not do this, your jobs will run in QSYSWRK.

All of the OS/400 servers that are supported by the sockets communications interface support prestart jobs. These servers are:

- Network print server
- Remote command and distributed program call server
- Central server
- Database server
- Secure Database server
- File server
- Secure File server
- Data Queue server
- Signon server (unique to servers using sockets communications support)

The following lists provides each of the prestart job entry attributes, and the initial values that are configured for the host servers using sockets communications support.

Subsystem Description

The subsystem that contains the prestart job entries.

OS/400 Server	Value
Network Print	QUSRWRK
Remote CMD/PGM Call	QUSRWRK
Central	QUSRWRK
Database	QSERVER
Secure Database	QSERVER
File	QSERVER
Secure File	QSERVER
Data Queue	QUSRWRK
Signon	QUSRWRK

Program library/name

The program that is called when the prestart job is started.

OS/400 Server	Value
Network Print	QIWS/QNPSERVS
Remote CMD/PGM Call	QIWS/QZRCSRVS
Central	QIWS/QZSCSRVS
Database	QIWS/QZDASOINIT
Secure Database	QIWS/QZDASSINIT
File	QSYS/QPWFSERVSO

OS/400 Server	Value
Secure File	QSYS/QPWFSERVSS
Data Queue	QIWS/QZHQSSRV
Signon	QIWS/QZSOSIGN

User profile

The user profile that the job runs under. This is what the job shows as the user profile. When a request to start a server is received from a client, the prestart job function switches to the user profile that is received in that request.

OS/400 Server	Value
Network Print	QUSER
Remote CMD/PGM Call	QUSER
Central	QUSER
Database	QUSER
Secure Database	QUSER
File	QUSER
Secure File	QUSER
Data Queue	QUSER
Signon	QUSER

Job name

The name of the job when it is started.

OS/400 Server	Value
Network Print	*PGM
Remote CMD/PGM Call	*PGM
Central	*PGM
Database	*PGM
Secure Database	*PGM
File	*PGM
Secure File	*PGM
Data Queue	*PGM
Signon	*PGM

Job description

The job description used for the prestart job. Note that if *USRPRF is specified, the job description for the profile that this job runs under will be used. This means QUSER's job description will be used. Some attributes from the requesting user's job description are also used; for example, print device and output queue are swapped from the requesting user's job description.

OS/400 Server	Value
Network Print	QSYS/QZBSJOBD
Remote CMD/PGM Call	QSYS/QZBSJOBD

OS/400 Server	Value
Central	QSYS/QZBSJOBD
Database	QSYS/QZBSJOBD
Secure Database	QSYS/QZBSJOBD
File	*USRPRF
Secure File	*USRPRF
Data Queue	QSYS/QZBSJOBD
Signon	QSYS/QZBSJOBD

Start jobs

Indicates whether prestart jobs are to automatically start when the subsystem is started. These prestart job entries are shipped with a start jobs value of *NO to prevent unnecessary jobs starting when you IPL. You can change this value to *YES if the host servers are used with sockets communications support. The STRHOSTSVR command starts each prestart job as part of its processing.

OS/400 Server	Value
Network Print	*NO
Remote CMD/PGM Call	*NO
Central	*NO
Database	*NO
Secure Database	*NO
File	*NO
Secure File	*NO
Data Queue	*NO
Signon	*NO

Initial number of jobs

The number of jobs that are started when the subsystem starts. This value is adjustable to suit your particular environment and needs.

OS/400 Server	Value
Network Print	1
Remote CMD/PGM Call	1
Central	1
Database	1
Secure Database	1
File	1
Secure File	1
Data Queue	1
Signon	1

Threshold

The minimum number of available prestart jobs for a prestart job entry. When this threshold is reached, additional prestart jobs automatically start. Threshold

maintains a certain number of jobs in the pool.

OS/400 Server	Value
Network Print	1
Remote CMD/PGM Call	1
Central	1
Database	1
Secure Database	1
File	1
Secure File	1
Data Queue	1
Signon	1

Additional number of jobs

The number of additional prestart jobs that are started when the threshold is reached.

OS/400 Server	Value
Network Print	2
Remote CMD/PGM Call	2
Central	2
Database	2
Secure Database	2
File	2
Secure File	2
Data Queue	2
Signon	2

Maximum number of jobs

The maximum number of prestart jobs that can be active for this entry.

OS/400 Server	Value
Network Print	*NOMAX
Remote CMD/PGM Call	*NOMAX
Central	*NOMAX
Database	*NOMAX
Secure Database	*NOMAX
File	*NOMAX
Secure File	*NOMAX
Data Queue	*NOMAX
Signon	*NOMAX

Maximum number of uses

The maximum number of uses of the job. A value of 200 indicates that the prestart job will end after 200 requests to start the server have been processed.

Note: The database server does not reuse any of the prestart jobs, even if this value is set to a value greater than one.

OS/400 Server	Value
Network Print	200
Remote CMD/PGM Call	1
Central	200
Database	1
Secure Database	200
File	*NOMAX
Secure File	*NOMAX
Data Queue	200
Signon	200

Wait for job

This causes a client connection request to wait for an available server job if the maximum number of jobs has been reached.

OS/400 Server	Value
Network Print	*YES
Remote CMD/PGM Call	*YES
Central	*YES
Database	*YES
Secure Database	*YES
File	*YES
Secure File	*YES
Data Queue	*YES
Signon	*YES

Pool identifier

The subsystem pool identifier in which this prestart job runs.

OS/400 Server	Value
Network Print	1
Remote CMD/PGM Call	1
Central	1
Database	1
Secure Database	1
File	1
Secure File	1
Data Queue	1
Signon	1

Class

The name and library of the class the prestart job runs under.

OS/400 Server	Value
Network Print	QGPL/QCASERVR
Remote CMD/PGM Call	QGPL/QCASERVR
Central	QGPL/QCASERVR
Database	QSYS/QPWFSERVER
Secure Database	QSYS/QPWFSERVER
File	QSYS/QPWFSERVER
Secure File	QSYS/QPWFSERVER
Data Queue	QGPL/QCASERVR
Signon	QGPL/QCASERVR

When the start jobs value for the prestart job entry has been set to *YES, and the remaining values are at their initial settings, the following actions take place for each prestart job entry:

- When the subsystem is started, one prestart job for each server is started.
- When the first client connection request processes for a specific server, the initial job is used and the threshold is exceeded.
- Additional jobs are started for that server based on the number that is defined in the prestart job entry.
- The number of available jobs is always at least one.
- The subsystem periodically checks the number of prestart jobs that are ready to process requests, and ends excess jobs. The subsystem always leaves at least the number of prestart jobs specified in the initial jobs parameter.

Monitoring Prestart Jobs

Use the Display Active Prestart Jobs (DSPACTPJ) command to monitor the prestart jobs. For example, to monitor prestart jobs for the signon server, you must know the subsystem your prestart jobs are in (QUSRWRK or a user-defined subsystem) and the program (for example, QZSOSIGN).

The DSPACTPJ command provides the following information:

```
                  Display Active Prestart Jobs                 AS400597
                                                        01/12/95  16:39:25
Subsystem  . . . . . :   QSYSWRK       Reset date . . . . . :   01/11/95
Program  . . . . . . :   QZSOSIGN      Reset time . . . . . :   16:54:50
  Library  . . . . . :     QIWS      Elapsed time . . . . :   0023:12:21

 Prestart jobs:
   Current number . . . . . . . . . . . . . . . . . :   10
   Average number . . . . . . . . . . . . . . . . . :   8.5
   Peak number  . . . . . . . . . . . . . . . . . . :   25

 Prestart jobs in use:
   Current number . . . . . . . . . . . . . . . . . :   5
   Average number . . . . . . . . . . . . . . . . . :   4.3
   Peak number  . . . . . . . . . . . . . . . . . . :   25


                                                                More...
```

```
                                                    01/12/95  16:39:25
Subsystem  . . . . . :   QSYSWRK       Reset date . . . . . :   01/11/95
Program  . . . . . :   QZSOSIGN      Reset time . . . . . :   16:54:50
  Library  . . . . . :     QIWS      Elapsed time . . . . :   0023:12:21



 Program start requests:
   Current number waiting . . . . . :   0
   Average number waiting . . . . . :   .2
   Peak number waiting  . . . . . :   4
   Average wait time  . . . . . :   00:00:20.0
   Number accepted  . . . . . :   0
   Number rejected  . . . . . :   0



                                                              Bottom
Press Enter to continue.

F3=Exit   F5=Refresh   F12=Cancel   F13=Reset statistics
```

Managing Prestart Jobs

Pressing the **F5** key while on the Display Active Prestart Jobs display can refresh the information presented for an active prestart job. The information about program start requests can indicate whether you need to change the available number of prestart jobs. If the information indicates that program start requests are waiting for an available prestart job, you can change prestart jobs with the Change Prestart Job Entry (CHGPJE) command.

If the program start requests are not acted on quickly, you can do any combination of the following:

- Increase the threshold
- Increase the parameter value for the initial number of jobs (INLJOBS)
- Increase the parameter value for the additional number of jobs (ADLJOBS)

The key is to ensure an available prestart job exists for every request.

Removing Prestart Job Entries

If you decide that you do not want the servers to use the prestart job function, you must do the following:

1. End the prestarted jobs with the End Prestart Job (ENDPJ) command.

   Prestarted jobs ended with the ENDPJ command are started the next time the subsystem is started if start jobs *YES is specified in the prestart job entry or when the STRHOSTSVR command is issued for the specified server type. If you only end the prestart job and don't take the next step, any requests to start the particular server will fail.
2. Remove the prestart job entries in the subsystem description with the Remove Prestart Job Entry (RMVPJE) command.

   The prestart job entries that are removed with the RMVPJE command are permanently removed from the subsystem description. Once the entry is removed, new requests for the server will succeed.

Routing Entries

When a daemon job is routed to a subsystem, the job is using the routing entries in the subsystem description. The routing entries for the host server daemon jobs are added to the subsystem description when the STRHOSTSVR command is issued. These jobs are started under the QUSER user profile. For daemon jobs that are submitted to the QSYSWRK subsystem, the QSYSNOMAX job queue is used. For daemon jobs that are submitted to the QSERVER subsystem, the QPWFSERVER job queue is used.

The server jobs run in the same subsystem as their corresponding daemon job. The characteristics of the server jobs are taken from their prestart job entry. If prestart jobs are not used for the servers, then the server jobs start with the characteristics of their corresponding daemon job.

The following information provides the initial configuration in the IBM-supplied subsystems for each of the server daemon jobs.

Network Print Server daemon

Subsystem	QSYS/QSYSWRK
Job Queue	QSYSNOMAX
User	QUSER
Routing Data	QNPSERVD
Job Name	QNPSERVD
Class	QGPL/QCASERVR
Sequence Number	2538

Remote Cmd/Pgm Call Server daemon

Subsystem	QSYS/QSYSWRK
Job Queue	QSYSNOMAX
User	QUSER
Routing Data	QZRCSRVSD
Job Name	QZRCSRVSD
Class	QGPL/QCASERVR
Sequence Number	2539

Central Server daemon

Subsystem	QSYS/QSYSWRK
Job Queue	QSYSNOMAX
User	QUSER
Routing Data	QZSCSRVSD
Job Name	QZSCSRVSD
Class	QGPL/QCASERVR
Sequence Number	2536

Database Server daemon

Subsystem	QSYS/QSERVER

Job Queue	QPWFSERVER
User	QUSER
Routing Data	QZDASRVSD
Job Name	QZDASRVSD
Class	QSYS/QPWFSERVER
Sequence Number	600

File Server daemon

Subsystem	QSYS/QSERVER
Job Queue	QPWFSERVER
User	QUSER
Routing Data	QPWFSERVSD
Job Name	QPWFSERVSD
Class	QSYS/QPWFSERVER
Sequence Number	200

Data Queue Server daemon

Subsystem	QSYS/QSYSWRK
Job Queue	QSYSNOMAX
User	QUSER
Routing Data	QZHQSRVD
Job Name	QZHQSRVD
Class	QGPL/QCASERVR
Sequence Number	2537

Signon Server daemon

Subsystem	QSYS/QSYSWRK
Job Queue	QSYSNOMAX
User	QUSER
Routing Data	QZSOSGND
Job Name	QZSOSGND
Class	QGPL/QCASERVR
Sequence Number	2540

Server Mapper daemon

Subsystem	QSYS/QSYSWRK
Job Queue	QSYSNOMAX
User	QUSER
Routing Data	QZSOSMAPD
Job Name	QZSOSMAPD
Class	QGPL/QCASERVR
Sequence Number	2541

Identifying Server Jobs on the AS/400

You may find using an emulator or green screen interface makes it difficult to relate a job to a certain personal computer or an individual client function. Being able to identify a particular job is a prerequisite to investigating problems and determining performance implications. In V4R4, you can use the Operations Navigator interface to identify your server jobs.

1. Double-click on the **AS/400 Operations Navigator** icon.
2. Open **Network** by clicking on the **+**.
3. Open **Servers** by clicking on the **+**.
4. Select the type of servers that you want to see jobs for (TCP/IP, Client Access etc.).
5. When the servers show in the right pane, right-click on the server that you want to see jobs for and click **Server Jobs**. Another window opens, showing the server jobs with the user, job type, job status, time entered system and date entered system for that server.

The following section provides information on how to identify server jobs using the traditional AS/400 green screen interface.

AS/400 Job Names

The job name that is used on the AS/400 consists of three parts:

- The simple job name
- The user ID
- The job number (ascending order)

The server jobs follow several conventions:

- Job name
  - For non-prestarted jobs, the server job name is the name of the server program.
  - Prestarted jobs use the name that is defined in the prestart job entry.
  - Jobs that are started by the servers use the job description name or a given name if they are batch jobs (the file server does this).
- The user ID
  - Is always QUSER, regardless of whether prestart jobs are used.
  - The job log shows which users have used the job.
- Work management creates the job number.

Displaying Server Jobs

Two methods can be used to identify server jobs. The first method is to use the WRKACTJOB command. The second method is to display the history log to determine which job is being used by which client.

Displaying Active Jobs with WRKACTJOB

The WRKACTJOB command shows all active jobs, as well as the server daemons and the server mapper daemon.

The following figures show a sample status with the WRKACTJOB command. Only jobs related to the servers are shown in the figures. You must press **F14** to see the available prestart jobs.

The following types of jobs are shown in the figures:

- **1** - Server mapper daemon
- **2** - Server daemons
- **3** - Prestarted server jobs

```
                          Work with Active Jobs                AS400597
                                                         01/12/95  10:25:40
CPU %:   3.1   Elapsed time:  21:38:40  Active jobs:  77

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
               .
___   QSYSWRK       QSYS       SBS      .0                  DEQW
___ 1 QZSOSMAPD     QUSER      BCH      .0                  SELW
               .
___ 2 QZSOSGND      QUSER      BCH      .0                  SELW
___   QZSCSRVSD     QUSER      BCH      .0                  SELW
___   QZRCSRVSD     QUSER      BCH      .0                  SELW
___   QZHQSRVD      QUSER      BCH      .0                  SELW
___   QNPSERVD      QUSER      BCH      .0                  SELW
               .
               .
___ 3 QZSOSIGN      QUSER      PJ       .0                  PSRW
___   QZSCSRVS      QUSER      PJ       .0                  PSRW
___   QZRCSRVS      QUSER      PJ       .0                  PSRW
___   QZHQSSRV      QUSER      PJ       .0                  PSRW
___   QNPSERVS      QUSER      PJ       .0                  PSRW
               .                                                  More...
```

```
                          Work with Active Jobs                AS400597
                                                         01/12/95  10:25:40
CPU %:   3.1   Elapsed time:  21:38:40  Active jobs:  77

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
               .
___    QSERVER      QSYS       SBS      .0                  DEQW
         QSERVER    QPGMR      ASJ      .1                  EVTW
               .
___ 2 QPWFSERVSD    QUSER      BCH      .0                  SELW
      QZDASRVSD     QUSER      BCH      .0                  SELW
               .
               .
___ 3 QPWFSERVSO    QUSER      PJ       .0                  PSRW
___   QPWFSERVSO    QUSER      PJ       .0                  PSRW
___   QZDASOINIT    QUSER      PJ       .0                  PSRW
___   QZDASOINIT    QUSER      PJ       .0                  PSRW
               .
               .                                                  More...
```

The following types of jobs are shown:

ASJ The autostart job for the subsystem

PJ The prestarted server jobs

SBS The subsystem monitor jobs

BCH The server daemon and the server mapper daemon jobs

Displaying the History Log

Each time a client user successfully connects to a server job, that job is swapped to run under the profile of that client user. To determine which job is associated with a particular client user, you can display the history log with the DSPLOG command. An example of the information that is provided is shown in the following figure.

```
                    Display History Log Contents
     .
     .
*SIGNON server job 056866/QUSER/QZSOSIGN processing request for user DMK on 0
     .
     .
Servicing job 056932/QUSER/QZSCSRVS for user KAREN on 01/12/95 08:23:15 in subs
     .
Servicing job 056980/QUSER/QZRCSRVS for user NANCY on 01/12/95 09:55:15 in su
     .
Servicing job 057011/QUSER/QZHQSSRV for user ROD on 01/12/95 10:01:42 in subs
     .
Servicing job 057187/QUSER/QNPSERVS for user KEVIN on 01/12/95 10:13:54 in s
     .
Servicing job 057201/QUSER/QPWFSERVSO for user DAVE on 01/12/95 10:15:01 in
     .
Servicing job 057299/QUSER/QZDASOINIT for user JOHN on 01/12/95 10:21:43 in
     .
Servicing job 056932/QUSER/QZSCSRVS for user KATHY on 01/12/95 11:17:20 in
     .
     .


Press Enter to continue.

F3=Exit   F10=Display all   F12=Cancel
```

Displaying server jobs for a user

To display the server jobs for a particular user,

1. Open **Operations Navigator** (double-click on the icon).
2. Click on **Users and Groups**, then **All Users**.
3. Right-click on the user that you want to see server jobs for.
4. Select **User Objects**, then click on **Jobs**. You see a window displaying all the server jobs for that user.

You can also use the WRKOBJLCK command. Specify the user profile and *USRPRF.

Chapter 5. Using Exit Programs

Exit programs allow system administrators to control which activities a client user is allowed for each of the specific servers. All of the servers support user-written exit programs. This chapter describes how the exit programs can be used, and how to configure them. It also provides sample programs that can help control access to server functions.

Registering Exit Programs

In order for the servers to know which exit program, if any, to call, you must register your exit program. You can register the exit program using the OS/400 registration facility.

Working with the Registration Facility

To register an exit program with the registration facility, use the Work with Registration Information (WRKREGINF) command.

```
                  Work with Registration Info (WRKREGINF)

Type choices, press Enter.

Exit point . . . . . .   *REGISTERED
Exit point format  . . . . . .   *ALL          Name, generic*, *ALL
Output . . . . . .   *            *, *PRINT
```

Press Enter to view the registered exit points.

```
                     Work with Registration Information

Type options, press Enter.
  5=Display exit point   8=Work with exit programs

                           Exit
     Exit                  Point
Opt  Point                 Format    Registered  Text
 _   QIBM_QGW_NJEOUTBOUND  NJEO0100     *YES     Network Job Entry outb
 8   QIBM_QHQ_DTAQ         DTAQ0100     *YES     Original Data Queue Se
 _   QIBM_QLZP_LICENSE     LICM0100     *YES     Original License Mgmt
 _   QIBM_QMF_MESSAGE      MESS0100     *YES     Original Message Serve
 _   QIBM_QNPS_ENTRY       ENTR0100     *YES     Network Print Server -
 _   QIBM_QNPS_SPLF        SPLF0100     *YES     Network Print Server -
 _   QIBM_QNS_CRADDACT     ADDA0100     *YES     Add CRQ description ac
 _   QIBM_QNS_CRCHGACT     CHGA0100     *YES     Change CRQ description
 _   QIBM_QNS_CRDLTSBMCRQ  DLTA0100     *YES     Delete submitted CRQ
 _   QIBM_QNS_CRDSPACT     DSPA0100     *YES     Display CRQ description
 _   QIBM_QNS_CREXCACT     EXCA0100     *YES     Run CRQ activity

Command
===>
```

Choose option 8 to work with the exit programs for the exit point defined for the server you would like to work with.

```
                          Work with Exit Programs

Exit point:   QIBM_QHQ_DTAQ            Format:   DTAQ0100

Type options, press Enter.
  1=Add   4=Remove   5=Display   10=Replace

              Exit
            Program     Exit
Opt          Number     Program        Library
1_                      __________     __________

  (No exit programs found)
```

Use option 1 to add an exit program to an exit point.

Notes:

1. If an exit program is already defined, you must remove it before you can change the name of the program.
2. Even though the registration facility can support multiple user exits for a specific exit point and format name, the servers always retrieve exit program 1.

```
                        Add Exit Program (ADDEXITPGM)

 Type choices, press Enter.

Exit point . . . . . > QIBM_QHQ_DTAQ
Exit point format  . . . . . > DTAQ0100      Name
Program number . . . . . > 1           1-2147483647, *LOW, *HIGH
 Program  . . . . . .   MYPGM          Name
   Library  . . . . . .     MYLIB  Name, *CURLIB
 Text 'description' . . . . . .   *BLANK
```

Enter your program name and library for the program at this exit point.

The same program is usable for multiple exit points. The program can use the data that is sent as input to determine how to handle different types of requests.

The following provides the exit point and format names for each of the specific OS/400 servers.

QIBM_QPWFS_FILE_SERV (File Server)

Format Name	PWFS0100
Application Name	*FILESRV

QIBM_QZDA_INIT (Database server initiation)

Format Name	ZDAI0100
Application Name	*SQL

QIBM_QZDA_NDB1 (Database server-native database requests)

Format Names	ZDAD0100 ZDAD0200
Application Name	*NDB

QIBM_QZDA_SQL1 (Database server SQL requests)

Format Names	ZDAQ0100 ZDAQ0200
Application Name	*SQLSRV

QIBM_QZDA_ROI1 (Database server retrieve object information requests)

Format Names	ZDAR0100 ZDAR0200
Application Name	*RTVOBJINF

QIBM_QZHQ_DATA_QUEUE (Data queue server)

Format Name	ZHQ00100
Application Name	*DATAQSRV

QIBM_QNPS_ENTRY (Network print server)

Format Name	ENTR0100
Application Name	QNPSERVR

QIBM_QNPS_SPLF (Network print server)

Format Name	SPLF0100
Application Name	QNPSERVR

QIBM_QZSC_LM (Central server license management requests)

Format Name	ZSCL0100
Application Name	*CNTRLSRV

QIBM_QZSC_NLS (Central server NLS requests)

Format Name	ZSCN0100
Application Name	*CNTRLSRV

QIBM_QZRC_RMT (Remote command and distributed program call server)

Format Name	CZRC0100
Application Name	*RMTSRV

QIBM_QZSO_SIGNONSRV (Signon server)

Format Name	ZSOY0100
Application Name	*SIGNON

How to Write Exit Programs

When you specify an exit program, either in the PCSACC network attribute or in the registration facility, the servers pass the following two parameters to the exit program before running your request:

- A 1-byte return code value
- A structure containing information about your request (This structure is different for each of the exit points.)

These two parameters allow the exit program to determine whether your request is possible. If the exit program sets the return code to X'F1', the server allows the request. If the return code is anything else, the server rejects the request.

For multiple servers and exit points, the same program is usable. The program can determine which server is being called and which function is being used by looking at the data in the second parameter structure.

"Appendix A. Parameter Formats for Exit Programs" on page 59 documents the structures of the second parameter that is sent to the exit programs. You can use this information to write your own exit programs.

Sample User Exit Programs

The sample exit programs in this section do not show all possible programming considerations or techniques, but you can review the examples before you begin your own design and coding.

Creating User Exit Programs with RPG/400

The following example illustrates how to set up a user exit program with RPG/400*.

```
     **
      ** OS/400 SERVERS - SAMPLE USER EXIT PROGRAM
      **
      ** THE FOLLOWING RPG/400 PROGRAM UNCONDITIONALLY
      ** ACCEPTS ALL REQUESTS. IT CAN BE USED AS A SHELL
      ** FOR SPECIFIC APPLICATIONS. NOTE: REMOVE THE
      ** SUBROUTINES AND CASE STATEMENT ENTRIES FOR THE SERVERS
      ** THAT DO NOT REQUIRE
      ** SPECIFIC EXIT PROGRAM HANDLING FOR BETTER PERFORMANCE.
      **
     E*
     E* NECESSARY ARRAY DEFINITIONS FOR TRANSFER FUNCTION
     E* AND REMOTE SQL
     E*
     E                    TFREQ    4096  1
     E                    RSREQ    4107  1
     I*
     I*
     IPCSDTA      DS
     I                                       1  10 USERID
     I                                      11  20 APPLID
     I*
     I* SPECIFIC PARAMETERS FOR VIRTUAL PRINTER
     I*
     I                                      21  30 VPFUNC
     I                                      31  40 VPOBJ
     I                                      41  50 VPLIB
     I                                      71  750VPIFN
     I                                      76  85 VPOUTQ
     I                                      86  95 VPQLIB
     I*
     I* SPECIFIC PARAMETERS FOR MESSAGING FUNCTION
     I                                      21  30 MFFUNC
     I*
     I* SPECIFIC PARAMETERS FOR TRANSFER FUNCTION
     I*
     I                                      21  30 TFFUNC
     I                                      31  40 TFOBJ
     I                                      41  50 TFLIB
```

```
I                                       51  60 TFMBR
I                                       61  70 TFFMT
I                                       71  750TFLEN
I                                       764171 TFREQ
I*
I* SPECIFIC PARAMETERS FOR FILE SERVER
I*
I* NOTE: FSNAME MAY BE UP TO l6MB.
I* FSNLEN WILL CONTAIN THE ACTUAL SIZE OF FSNAME.
I*
I                                     B  21  240FSFID
I                                        25  32 FSFMT
I                                        33  33 FSREAD
I                                        34  34 FSWRIT
I                                        35  35 FSRDWR
I                                        36  36 FSDLT
I                                     B  37  400FSNLEN
I                                        41 296 FSNAME
I*
I* SPECIFIC PARAMETERS FOR DATA QUEUES
I*
I                                       21  30 DQFUNC
I                                       31  40 DQQ
I                                       41  50 DQLIB
I                                       70  750DQLEN
I                                       76  77 DQROP
I                                       78  820DQKLEN
I                                       83 338 DQKEY
I*
I* SPECIFIC PARAMETERS FOR REMOTE SQL
I*
I                                       21  30 RSFUNC
I                                       31  40 RSOBJ
I                                       41  50 RSLIB
I                                       51  51 RSCMT
I                                       52  52 RSMODE
I                                       53  53 RSCID
I                                       54  71 RSSTN
I                                       72  75 RSRSV
I                                       764182 RSREQ
I*
I* SPECIFIC PARAMETERS FOR NETWORK PRINT SERVER
I*
I                                       21  28 NPFT
I                                     B  29  320NPFID
I* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT SPLF0l00
I                                       33  42 NPJOBN
I                                       43  52 NPUSRN
I                                       53  58 NPJOB#
I                                       59  68 NPFILE
I                                     B  69  720NPFIL#
I                                     B  73  760NPLEN
I                                        77 332 NPDATA
I*
I* Data Queue server:
I*
I* QIBM_QZHQ_DATA_QUEUE  format ZHQ00100
I*
I                                       21  28 DQOFMT
I                                     B  29  320DQOFID
I                                       33  42 DQOOBJ
I                                       43  52 DQOLIB
I                                       53  54 DQOROP
I                                     B  55  580DQOLEN
I                                       59 314 DQOKEY
I*
I* Specific PARAMETERS FOR CENTRAL SERVER
```

```
I*
I                                        21  28 CSFMT
I                                     B  29  320CSFID
I* Central server:
I*
I* QIBM_QZSC_LM format ZSCL0l00 for license management calls
I*
I*
I                                        33 287 CSLCNM
I                                       288 295 CSLUSR
I                                       296 302 CSLPID
I                                       303 306 CSLFID
I                                       307 312 CSLRID
I                                     B 313 3140CSLTYP
I*
I* Central server:
I*
I* QIBM_QZSC_LM format ZSCS0l00 for system management calls
I*
I*
I                                        33 287 CSSCNM
I                                       288 542 CSSCMY
I                                       543 543 CSSNDE
I                                       544 798 CSSNNM
I*

I* Central server:
I*
I* QIBM_QZSC_LM format ZSCN0l00 for retrive conversion map calls
I*
I*
I                                        21  30 CSNXFM
I                                        29  320CSNFNC
I                                     B  33  360CSNFRM
I                                     B  37  400CSNTO
I                                     B  41  420CSNCNT
I*
I* SPEClFIC PARAMETERS FOR DATABASE SERVER
I*
I                                        21  28 DBFMT
I                                     B  29  320DBFID
I*
I* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZDAD0l00
I                                        33 160 DBDFIL
I                                       161 170 DBDLIB
I                                       171 180 DBDMBR
I                                       181 190 DBDAUT
I                                       191 318 DBDBFL
I                                       319 328 DBDBLB
I                                       329 338 DBDOFL
I                                       339 348 DBDOLB
I                                       349 358 DBDOMB
I*
I* THE FOLLOWING PARRMETERS ADDITIONAL FOR FORMAT ZDAD0200
I                                     B  33  360DBNUM
I                                        37  46 DBLIB2
I*
I* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZDAQ0l00
I                                        33  50 DBSTMT
I                                        51  68 DBCRSR
I                                        69  70 DBOPI
I                                        71  72 DBATTR
I                                        73  82 DBPKG
I                                        83  92 DBPLIB
I                                     B  93  940DBDRDA
I                                        95  95 DBCMT
I                                        96 351 DBTEXT
```

```
I* THE FOLLOWING PARAMETERS REPLACE DBTEXT FOR FORMAT ZDAQ0200
I                                        96 105 DBSQCL
I                                     B 133 1360DBSQLN
I                                       137 392 DBSQTX
I* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZDAR0l00
I                                        33  52 DBLIBR
I                                        53  88 DBRDBN
I                                        89 108 DBPKGR
I                                       109 364 DBFILR
I                                       365 384 DBMBRR
I                                       385 404 DBFFT

I* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZDAR0200
I                                        33  42 DBRPLB
I                                        43 170 DBRPTB
I                                       171 180 DBRFLB
I                                       181 308 DBRFTB
I*
I* Remote Command/Distributed Program Call server:
I*
I* QIBM_QZRC_RMT  format CZRC0100
I*    RCPGM AND RCLIB ARE NOT USED FOR REMOTE COMMAND CALLS
I*
I                                        21  28 RCFMT
I                                     B  29  320RCFID
I                                        33  42 RCPGM
I                                        43  52 RCLIB
I                                     B  53  560RCNUM
I                                        57 312 RCDATA
I*
I* Signon server:
I*
I* QIBM_QZSO_SIGNONSRV format ZSOY0l00 for TCP/IP signon server
I*
I                                        21  28 SOXFMT
I                                     B  29  320SOFID
I*
I*************************************************************
I*
I              '*VPRT     '          C         #VPRT
I              '*TFRFCL   '          C         #TRFCL
I              '*FILESRV  '          C         #FILE
I              '*MSGFCL   '          C         #MSGF
I              '*DQSRV    '          C         #DQSRV
I              '*RQSRV    '          C         #RQSRV
I              '*SQL      '          C         #SQL
I              '*NDB      '          C         #NDBSV
I              '*SQLSRV   '          C         #SQLSV
I              '*RTVOBJINF'          C         #RTVOB
I              '*DATAQSRV '          C         #DATAQ
I              '*QNPSERVR '          C         #QNPSV
I              '*CNTRLSRV '          C         #CNTRL
I              '*RMTSRV   '          C         #RMTSV
I              '*SIGNON  '          C         #SIGN
I*
C*
C* EXIT PROGRAM CALL PARAMETERS
C*
C           *ENTRY    PLIST
C                     PARM           RTNCD   1
C                     PARM           PCSDTA
C*

C* INITIALIZE RETURN VALUE TO ACCEPT REQUEST
C*
C                     MOVE '1'       RTNCD
```

```
C*
C* COMMON PROCESSING
C*
C*              COMMON LOGIC GOES HERE
C*
C* PROCESS BASED ON SERVER ID
C*
C           APPLID    CASEQ#VPRT     VPRT
C           APPLID    CASEQ#TRFCL    TFR
C           APPLID    CASEQ#FILE     FILE
C           APPLID    CASEQ#MSGF     MSG
C           APPLID    CASEQ#DQSRV    DATAQ
C           APPLID    CASEQ#RQSRV    RSQL
C           APPLID    CASEQ#SQL      SQLINT
C           APPLID    CASEQ#NDBSV    NDB
C           APPLID    CASEQ#SQLSV    SQLSRV
C           APPLID    CASEQ#RTVOB    RTVOBJ
C           APPLID    CASEQ#DATAQ    ODATAQ
C           APPLID    CASEQ#QNPSV    NETPRT
C           APPLID    CASEQ#CNTRL    CENTRL
C           APPLID    CASEQ#RMTSV    RMTCMD
C           APPLID    CASEQ#SIGN     SIGNON
C                     END
C                     SETON                    LR
C                     RETRN
C*
C* SUBROUTINES
C*
C*
C* VIRTUAL PRINT
C*
C           VPRT      BEGSR
C*            SPECIFIC LOGIC GOES HERE
C                     ENDSR
C*
C* TRANSFER FUNCTION
C*
C* THE FOLLOWING IS AN EXAMPLE OF SPECIFIC PROCESSING
C* THAT THE EXIT PROGRAM COULD DO FOR TRANSFER FUNCTION.
C*


C* IN THIS CASE, USERS ARE NOT ALLOWED TO SELECT
C* DATA FROM ANY FILES THAT ARE IN LIBRARY QIWS.
C*
C           TFR       BEGSR
C           TFFUNC    IFEQ 'SELECT'
C           TFLIB     ANDEQ'QIWS'
C                     MOVE '0'       RTNCD
C                     END
C                     ENDSR
C*
C*
C* FILE SERVER
C*
C           FILE      BEGSR
C*             SPECIFIC LOGIC GOES HERE
C                     ENDSR
C*
C* MESSAGING FUNCTION
C*
C           MSG       BEGSR
C*             SPECIFIC LOGIC GOFS HERE
C                     ENDSR
C* DATA QUEUES
C*
C           DATAQ     BEGSR
C*             SPECIFIC LOGIC GOES HERE
```

```
C                     ENDSR
C*
C* REMOTE SQL
C*
C           RSQL      BEGSR
C*             SPECIFIC LOGIC GOES HERE
C                     ENDSR
C*
C* SERVERS
C*
C*
C* DATABASE INIT
C*
C           SQLINT    BEGSR
C*             SPECIFIC LOGIC GOES HERE
C                     ENDSR
C*
C* DATABASE NDB (NATIVE DATABASE)
C*
C           NDB       BEGSR
C*             SFECIFIC LOGIC GOES HERE
C                     ENDSR
C*

C* DATABASE SQL
C*
C           SQLSRV    BEGSR
C*             SPECIFIC LOGIC GOES HERE
C                     ENDSR
C*
C* DATABASE RETRIVE OBJECT INFORMATION
C*
C           RTVOBJ    BEGSR
C*             SPECIFIC LOGIC GOES HERE
C                     ENDSR
C*
C* DATA QUEUE SERVER
C*
C           ODATAQ    BEGSR
C*             SPECIFIC LOGIC GOES HERE
C                     ENDSR
C*
C* NETWORK PRINT
C*
C           NETPRT    BEGSR
C*             SPECIFIC LOGIC GOES HERE
C                     ENDSR
C*
C* CENTRAL SERVER
C*
C*
C* THE FOLLOWING IS AN EXAMPLE OF SPECIFIC PROCESSING
C* THAT THE EXIT PROGRAM COULD DO FOR LICENSE MANAGEMENT.
C*
C* IN THIS CASE, THE USER "USERALL" WILL NOT BE ALLOWED
C* TO EXECUTE ANY FUNCTIONS THAT ARE PROVIDED BY THE
C* CENTRAL SERVER FOR WHICH THIS PROGRAM IS A REGISTERED
C* EXIT PROGRAM - LICENSE INFORMATION, SYSTEM MANAGEMENT
C* OR RETRIVE A CONVERSION MAP.
C*
C           CENTRL    BEGSR
C           USERID    IFEQ 'USERALL'
C                     MOVE '0'       RTNCD
C                     ENDIF
```

```
C*            SPECIFIC LOGIC GOES HERE
C                     ENDSR
C*

C* REMOTE COMMAND/DISTRIBUTED PROGRAM CALL
C*
C* IN THIS CASE, THE USER "USERALL" WILL NOT BE ALLOWED
C* TO EXECUTE ANY REMOTE COMMANDS OR REMOTE PROGRAM CALLS
C*
C           RMTCMD    BEGSR
C           USERID    IFEQ 'USERALL'
C                     MOVE '0'       RTNCD
C                     ENDIF
C                     ENDSR
C*
C* SIGNON SERVER
C*
C           SIGNON    BEGSR
C*                  SPECIFIC LOGIC GOES HERE
C                     ENDSR
```

Creating User Exit Programs with Control Language

The following example illustrates how to set up a user exit program control language (CL).

```
/*****************************************************************/
/*                                                               */
/* AS/400 SERVERS- SAMPLE USER EXIT PROGRAM                      */
/*                                                               */
/* THE FOLLOWING CONTROL LANGUAGE PROGRAM UNCONDITIONALLY        */
/* ACCEPTS ALL REQUESTS. IT CAN BE USED AS A SHELL FOR DEVELOPING */
/* EXIT PROGRAMS TAILORED FOR YOUR OPERATING ENVIRONMENT.        */
/*                                                               */
/*                                                               */
/*****************************************************************/
PGM PARM(&STATUS &REQUEST)

/* * * * * * * * * * * * * * * * * * * */
/*                                     */
/* PROGRAM CALL PARAMETER DECLARATIONS */
/*                                     */
/* * * * * * * * * * * * * * * * * * * */

DCL VAR(&STATUS) TYPE(*CHAR) LEN(1) /* Accept/Reject indicator */
/* */
/* Note: Request is declared as *CHAR LEN(2000) because that is */
/* the limit in CL. The actual length of REQUEST is 4171. */
/* */
DCL VAR(&REQUEST) TYPE(*CHAR) LEN(2000) /* Parameter structure */

/***********************************/
/*                                 */
/* PARAMETER DECLARES              */
/*                                 */
/***********************************/

/* COMMON DECLARES */
DCL VAR(&USER) TYPE(*CHAR) LEN(10)
/* User ID     */
DCL VAR(&APPLIC) TYPE(*CHAR) LEN(10)
/* Server ID   */
DCL VAR(&FUNCTN) TYPE(*CHAR) LEN(10) /* Function being performed   */
```

```
 /* VIRTUAL PRINT DECLARES */
DCL VAR(&VPOBJ)  TYPE(*CHAR) LEN(10)  /* Object name            */
DCL VAR(&VPLIB)  TYPE(*CHAR) LEN(10)  /* Object library name    */
DCL VAR(&VPLEN)  TYPE(*DEC) LEN(5 0)   /* Length of following fields*/
DCL VAR(&VPOUTQ) TYPE(*CHAR) LEN(10) /* Output queue name     */
DCL VAR(&VPQLIB) TYPE(*CHAR) LEN(10) /* Output queue library name */

/* TRANSFER FUNCTION DECLARES */
 DCL VAR(&TFOBJ) TYPE(*CHAR) LEN(10)   /* Object name */
 DCL VAR(&TFLIB) TYPE(*CHAR) LEN(10)   /* Object library name */
 DCL VAR(&TFMBR) TYPE(*CHAR) LEN(10)   /* Member name */
 DCL VAR(&TFFMT) TYPE(*CHAR) LEN(10)   /* Record format name */
 DCL VAR(&TFLEN) TYPE(*DEC) LEN(5 0)   /* Length of request */
 DCL VAR(&TFREQ) TYPE(*CHAR) LEN(1925) /*Transfer request statement*/

/* FILE SERVER DECLARES */
DCL VAR(&FSFID) TYPE(*CHAR) LEN(4)   /* Function identifier   */
DCL VAR(&FSFMT) TYPE(*CHAR) LEN(8)   /* Parameter format      */
DCL VAR(&FSREAD) TYPE(*CHAR) LEN(1)  /* Open for read         */
DCL VAR(&FSWRITE) TYPE(*CHAR) LEN(1) /* Open for write        */
DCL VAR(&FSRDWRT) TYPE(*CHAR) LEN(1) /* Open for read/write   */
DCL VAR(&FSDLT) TYPE(*CHAR) LEN(1)   /* Open for delete       */
DCL VAR(&FSLEN) TYPE(*CHAR) LEN(4)   /* fname length          */
DCL VAR(&FSNAME) TYPE(*CHAR) LEN(2000) /* Qualified file name   */

/* DATA QUEUE DECLARES */
DCL VAR(&DQQ)    TYPE(*CHAR) LEN(10)  /* Data queue name */
DCL VAR(&DQLIB)  TYPE(*CHAR) LEN(10)  /* Data queue library name */
DCL VAR(&DQLEN)  TYPE(*DEC)  LEN(5 0) /* Total request length */
DCL VAR(&DQROP)  TYPE(*CHAR) LEN(2)   /* Relational operator */
DCL VAR(&DQKLEN) TYPE(*DEC)  LEN(5 0) /* Key length */
DCL VAR(&DQKEY)  TYPE(*CHAR) LEN(256) /* Key value */

/* REMOTE SQL DECLARES */
DCL VAR(&RSOBJ) TYPE(*CHAR) LEN(10) /* Object name               */
DCL VAR(&RSLIB) TYPE(*CHAR) LEN(10) /* Object library name       */
DCL VAR(&RSCMT) TYPE(*CHAR) LEN(1) /* Commitment control level */
DCL VAR(&RSMODE) TYPE(*CHAR) LEN(1) /* Block/Update mode indicator*/
DCL VAR(&RSCID) TYPE(*CHAR) LEN(1) /* Cursor ID              */
DCL VAR(&RSSTN) TYPE(*CHAR) LEN(18) /* Statement name         */
DCL VAR(&RSRSU) TYPE(*CHAR) LEN(4) /* Reserved               */
DCL VAR(&RSREQ) TYPE(*CHAR) LEN(1925)/* SQL statement         */

/* NETWORK PRINT SERVER DECLARES */
DCL VAR(&NPFMT) TYPE(*CHAR) LEN(8) /* Format name            */
DCL VAR(&NPFID)      TYPE(*CHAR) LEN(4) /* Function identifier */
/* THE FOLLOWlING PARAMETERS ADDITIONAL FOR FORMAT SPLFOlOO */
DCL VAR(&NPJOBN)     TYPE(*CHAR) LEN(10)/* Job name             */
DCL VAR(&NPUSRN)     TYPE(*CHAR) LEN(10)/* User name            */
DCL VAR(&NPJOB#)     TYPE(*CHAR) LEN(6) /* Job number           */
DCL VAR(&NPFILE)     TYPE(*CHAR) LEN(10)/* File name            */
DCL VAR(&NPFIL#)     TYPE(*CHAR) LEN(4) /* File number          */
DCL VAR(&NPLEN)      TYPE(*CHAR) LEN(4) /* Data Length          */
DCL VAR(&NPDATA)     TYPE(*CHAR) LEN(2000) /* Data              */

DCL VAR(&DBNUM) TYPE(*CHAR) LEN(4) /* Number of libraries    */
DCL VAR(&DBLIB2) TYPE(*CHAR) LEN(10) /* Library name          */


/* DATA QUEUE SERVER DECLARES */
DCL VAR(&DQFMT)    TYPE(*CHAR) LEN(8)    /* Format name             */
DCL VAR(&DQFID)    TYPE(*CHAR) LEN(4)    /* Function IDENTIFIER  */
DCL VAR(&DQOOBJ)   TYPE(*CHAR) LEN(10)   /* Object name             */
DCL VAR(&DQOLIB)   TYPE(*CHAR) LEN(10)   /* Library name            */
DCL VAR(&DQOROP)   TYPE(*CHAR) LEN(2) /* Relational operator       */
DCL VAR(&DQOLEN)   TYPE(*CHAR) LEN(4) /* Key length               */
```

```
DCL VAR(&DQOKEY)   TYPE(*CHAR) LEN(256) /* Key                      */

/* CENTRAL SERVER DECLARES */
DCL VAR(&CSFMT)    TYPE(*CHAR) LEN(8)   /* Format name              */
DCL VAR(&CSFID)    TYPE(*CHAR) LEN(4) /* Function identifier        */
/* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZSCL0100 */
DCL VAR(&CSCNAM)   TYPE(*CHAR) LEN(255) /* Unique client name      */
DCL VAR(&CSLUSR)   TYPE(*CHAR) LEN(8)     /* License users handle    */
DCL VAR(&CSPID)    TYPE(*CHAR) LEN(7)      /* Product identification   */
DCL VAR(&CSFID)    TYPE(*CHAR) LEN(4)      /* Feature identification   */
DCL VAR(&CSRID)    TYPE(*CHAR) LEN(6)      /* Release identification   */
DCL VAR(&CSTYPE)   TYPE(*CHAR) LEN(2) /* Type of information req    */
/* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZSCS0100  */
DCL VAR(&CSCNAM)   TYPE(*CHAR) LEN(255) /* Unique client name       */
DCL VAR(&CSCMTY)   TYPE(*CHAR) LEN(255) /* Community name           */
DCL VAR(&CSNODE)   TYPE(*CHAR) LEN(1) /* Node type                  */
DCL VAR(&CSNNAM)   TYPE(*CHAR) LEN(255) /* Node name                */
/* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZSCN0100  */
DCL VAR(&CSFROM)   TYPE(*CHAR) LEN(4) /* From CCSID                 */
DCL VAR(&CSTO)     TYPE(*CHAR) LEN(4)   /* To CCSID                */
DCL VAR(&CSCTYP)   TYPE(*CHAR) LEN(2)     /* Type of conversion       */
/* DATABASE SERVER DECLARES */
DCL VAR(&DBFMT)    TYPE(*CHAR) LEN(8)   /* Format name              */
DCL VAR(&DBFID)    TYPE(*CHAR) LEN(4) /* Function identifier        */


/* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZDAD0100 */
DCL VAR(&DBFILE)   TYPE(*CHAR) LEN(128)   /* File name             */
DCL VAR(&DBLIB)    TYPE(*CHAR) LEN(10)    /* Library name           */
DCL VAR(&DBMBR)    TYPE(*CHAR) LEN(10)    /* Member name            */
DCL VAR(&DBAUT)    TYPE(*CHAR) LEN(10)    /* Authority to file      */
DCL VAR(&DBBFIL)   TYPE(*CHAR) LEN(128)   /* Based on file name     */
DCL VAR(&DBBLIB)   TYPE(*CHAR) LEN(10)   /* Based on library name   */
DCL VAR(&DBOFIL)   TYPE(*CHAR) LEN(10)   /* Override file name      */
DCL VAR(&DBOLIB)   TYPE(*CHAR) LEN(10)   /* Override library name   */
DCL VAR(&DBOMBR)   TYPE(*CHAR) LEN(10)   /* Override member name    */

/* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZDAD0200 */
 DCL VAR(&DBNUM)   TYPE(*CHAR) LEN(4) /* Number of libraries    */
 DCL VAR(&DBLIB2)  TYPE(*CHAR) LEN(10) /* Library name           */

/* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZDAQ0100 */
DCL VAR(&DBSTMT) TYPE(*CHAR) LEN(18) /* Statement name         */
DCL VAR(&DBCRSR) TYPE(*CHAR) LEN(18) /* Cursor name            */
DCL VAR(&DBOPT)  TYPE(*CHAR) LEN(2) /* Prepare option          */
DCL VAR(&DBATTR) TYPE(*CHAR) LEN(2) /* Open attributes        */
DCL VAR(&DBPKG)  TYPE(*CHAR) LEN(10) /* Package name           */
DCL VAR(&DBPLIB) TYPE(*CHAR) LEN(10) /* Package library name */
DCL VAR(&DBDRDA) TYPE(*CHAR) LEN(2) /* DRDA indicator         */
DCL VAR(&DBCMT)  TYPE(*CHAR) LEN(1)    /* Commit control level */
DCL VAR(&DBTEXT) TYPE(*CHAR) LEN(512) /* First 512 bytes of stmt */


/* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZDAR0100 */
DCL VAR(&DBLIBR) TYPE(*CHAR) LEN(20) /* Library name               */
DCL VAR(&DBRDBN) TYPE(*CHAR) LEN(36) /* Relational Database name */
DCL VAR(&DBPKGR) TYPE(*CHAR) LEN(20) /* Package name              */
DCL VAR(&DBFILR) TYPE(*CHAR) LEN(256) /* File name (SQL alias)    */
DCL VAR(&DBMBRR) TYPE(*CHAR) LEN(20) /* Member name               */
DCL VAR(&DBFFMT) TYPE(*CHAR) LEN(20) /* Format name               */

/* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZDAR0200 */
DCL VAR(&DBPLIB) TYPE(*CHAR) LEN(10) /* Primary key table lib      */
DCL VAR(&DBPTBL) TYPE(*CHAR) LEN(128) /* Primary key table        */
DCL VAR(&DBFLIB) TYPE(*CHAR) LEN(10) /* Foreign key table lib     */
DCL VAR(&DBFTBL) TYPE(*CHAR) LEN(128) /* Foreign key table        */
```

```
/* REMOTE COMMAND SERVER DECLARES */
DCL VAR(&RCFMT) TYPE(*CHAR) LEN(8) /* Format name                */
DCL VAR(&RCFID) TYPE(*CHAR) LEN(4) /* Function identifier        */
DCL VAR(&RCPGM) TYPE(*CHAR) LEN(10) /* Program name               */
DCL VAR(&RCLIB) TYPE(*CHAR) LEN(10) /* Program library name       */
DCL VAR(&RCNUM) TYPE(*CHAR) LEN(4) /* Number of parms or cmdlen */
DCL VAR(&RCDATA) TYPE(*CHAR) LEN(6000)/* Command string nor parms */

/* SIGNON SERVER DECLARES */

DCL VAR(&SOFMT) TYPE(*CHAR) LEN(8) /* Format name                */
DCL VAR(&SOFID) TYPE(*CHAR) LEN(4) /* Function identifier        */


/*********************************/
/*                               */
/* OTHER DECLARES                */
/*                               */
/*********************************/
 DCL VAR(&WRKLEN) TYPE(*CHAR) LEN(5)
 DCL VAR(&DECLEN) TYPE(*DEC) LEN(8 0)

/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
/*                                                               */
/* EXTRACT THE VARIOUS PARAMETERS FROM THE STRUCTURE  */
/*                                                               */
/* * * * * * * * * * * * * * * * * * * * * * * * * */

/* HEADER */
CHGVAR VAR(&USER)   VALUE(%SST(&REQUEST 1 10))
   CHGVAR VAR(&APPLIC) VALUE(%SST(&REQUEST 11 10))
   CHGVAR VAR(&FUNCTN) VALUE(%SST(&REQUEST 21 10))

/* VIRTUAL PRINTER */
   CHGVAR VAR(&VPOBJ)  VALUE(%SST(&REQUEST 31 10))
   CHGVAR VAR(&VPLIB)  VALUE(%SST(&REQUEST 41 10))
   CHGVAR VAR(&WRKLEN) VALUE(%SST(&REQUEST 71 5))
   CHGVAR VAR(&VPLEN)  VALUE(%BINARY(&WRKLEN 1 4))
   CHGVAR VAR(&VPOUTQ) VALUE(%SST(&REQUEST 76 10))
   CHGVAR VAR(&VPQLIB) VALUE(%SST(&REQUEST 86 10))


/* TRANSFER FUNCTION */
   CHGVAR VAR(&TFOBJ)  VALUE(%SST(&REQUEST 31 10))
   CHGVAR VAR(&TFLIB)  VALUE(%SST(&REQUEST 41 10))
   CHGVAR VAR(&TFMBR)  VALUE(%SST(&REQUEST 51 10))
   CHGVAR VAR(&TFFMT)  VALUE(%SST(&REQUEST 61 10))
   CHGVAR VAR(&WRKLEN) VALUE(%SST(&REQUEST 71 5))
   CHGVAR VAR(&TFLEN)  VALUE(%BINARY(&WRKLEN 1 4))
   CHGVAR VAR(&TFREQ)  VALUE(%SST(&REQUEST 76 1925))

/* FILE SERVER */
   CHGVAR VAR(&FSFID)      VALUE(%SST(&REQUEST   21   4))
   CHGVAR VAR(&FSFMT)      VALUE(%SST(&REQUEST   25   8))
   CHGVAR VAR(&FSREAD)     VALUE(%SST(&REQUEST   33   1))
   CHGVAR VAR(&FSWRITE)   VALUE(%SST(&REQUEST   34   1))
   CHGVAR VAR(&FSRDWRT)   VALUE(%SST(&REQUEST   35   1))
   CHGVAR VAR(&FSDLT)      VALUE(%SST(&REQUEST   36   1))
   CHGVAR VAR(&FSLEN)      VALUE(%SST(&REQUEST   37   4))
   CHGVAR VAR(&DECLEN)     VALUE(%BINARY(&FSLEN 1 4))
   CHGVAR VAR(&FSNAME)     VALUE(%SST(&REQUEST   41   &DECLEN))
```

```
/* DATA QUEUES */
   CHGVAR VAR(&DQQ)        VALUE(%SST(&REQUEST 31 10))
   CHGVAR VAR(&DQLIB)      VALUE(%SST(&REQUEST 41 10))
   CHGVAR VAR(&WRKLEN)     VALUE(%SST(&REQUEST 71  5))
   CHGVAR VAR(&DQLEN)      VALUE(%BINARY(&WRKLEN 1 4))
   CHGVAR VAR(&DQROP)      VALUE(%SST(&REQUEST 76  2))
   CHGVAR VAR(&WRKLEN)     VALUE(%SST(&REQUEST 78  5))
   CHGVAR VAR(&DQKLEN)     VALUE(&WRKLEN)
   CHGVAR VAR(&DQKEY)      VALUE(%SST(&REQUEST 83 &DQKLEN))


 /* REMOTE SQL */
   CHGVAR VAR(&RSOBJ)    VALUE(%SST(&REQUEST 31 10))
   CHGVAR VAR(&RSLIB)    VALUE(%SST(&REQUEST 41 10))
   CHGVAR VAR(&RSCMT)    VALUE(%SST(&REQUEST 51 1))
   CHGVAR VAR(&RSMODE)   VALUE(%SST(&REQUEST 52 1))
   CHGVAR VAR(&RSCID)    VALUE(%SST(&REQUEST 53 1))
   CHGVAR VAR(&RSSTN)    VALUE(%SST(&REQUEST 54 18))
   CHGVAR VAR(&RSRSU)    VALUE(%SST(&REQUEST 72 4))
   CHGVAR VAR(&RSREQ)    VALUE(%SST(&REQUEST 76 1925))

/* NETWORK PRINT SERVER */
   CHGVAR VAR(&NPFMT)    VALUE(%SST(&REQUEST 21 8))
   CHGVAR VAR(&NPFID)    VALUE(%SST(&REQUEST 29 4))

/* IF FORMAT IS SPLF0100 */
IF  COND(&NPFMT *EQ 'SPLF0100') THEN(DO)
   CHGVAR VAR(&NPJOBN)   VALUE(%SST(&REQUEST 33 10))
   CHGVAR VAR(&NPUSRN)   VALUE(%SST(&REQUEST 43 10))
   CHGVAR VAR(&NPJOB#)   VALUE(%SST(&REQUEST 53 6))
   CHGVAR VAR(&NPFILE)   VALUE(%SST(&REQUEST 59 10))
   CHGVAR VAR(&NPFIL#)   VALUE(%SST(&REQUEST 69 4))
   CHGVAR VAR(&NPLEN)    VALUE(%SST(&REQUEST 73 4))
   CHGVAR VAR(&DECLEN)   VALUE(%BINARY(&NPLEN 1 4))
   CHGVAR VAR(&NPDATA)   VALUE(%SST(&REQUEST 77 &DECLEN))
ENDDO


/* DATA QULUE SERVER */
   CHGVAR VAR(&DQFMT)  VALUE(%SST(&REQUEST 21 8))
   CHGVAR VAR(&DQFID)  VALUE(%SST(&REQUEST 29 4))
   CHGVAR VAR(&DQOOBJ) VALUE(%SST(&REQUEST 33 10))
   CHGVAR VAR(&DQOLIB) VALUE(%SST(&REQUEST 43 10))
   CHGVAR VAR(&DQOROP) VALUE(%SST(&REQUEST 53 2))
   CHGVAR VAR(&DQOLEN) VALUE(%SST(&REQUEST 55 4))
   CHGVAR VAR(&DQOKEY) VALUE(%SST(&REQUEST 59 256))

/* CENTRAL SERVER */
   CHGVAR VAR(&CSFMT) VALUE(%SST(&REQUEST  1 8))
   CHGVAR VAR(&CSFID) VALUE(%SST(&REQUEST 29 4))

/* IF FORMAT IS ZSCL0100 */
IF COND(&CSFMT *EQ 'ZSCL0100') THEN(DO)
   CHGVAR VAR(&CSCNAM) VALUE(%SST(&REQUEST 33 255))
   CHGVAR VAR(&CSLUSR)  VALUE(%SST(&REQUEST 288 8))
   CHGVAR VAR(&CSPID)   VALUE(%SST(&REQUEST 296 7))
   CHGVAR VAR(&CSFID)   VALUE(%SST(&REQUEST 303 4))
   CHGVAR VAR(&CSRID)   VALUE(%SST(&REQUEST 307 6))
   CHGVAR VAR(&CSTYPE)  VALUE(%SST(&REQUEST 313 2))
ENDDO


/* IF FORMAT IS ZSCS0100 */
IF COND(&CSFMT *EQ 'ZSCS0100') THEN(DO)
  CHGVAR VAR(&CSCNAM) VALUE(%SST(&REQUEST 33 255))
  CHGVAR VAR(&CSCMTY) VALUE(%SST(&REQUEST 288 255))
```

```
  CHGVAR VAR(&CSNODE) VALUE(%SST(&REQUEST 543 1))
  CHGVAR VAR(&CSNNAM) VALUE(%SST(&REQUEST 544 255))
  ENDDO


/* IF FORMAT IS ZSCN0100 */
IF COND(&CSFMT *EQ 'ZSCN0100') THEN(DO)
  CHGVAR VAR(&CSFROM) VALUE(%SST(&REQUEST 33 4))
  CHGVAR VAR(&CSTO)   VALUE(%SST(&REQUEST 37 4))
  CHGVAR VAR(&CSCTYP) VALUE(%SST(&REQUEST 41 2))
  ENDDO

/* DATABASE SERVER */
   CHGVAR VAR(&DBFMT)    VALUE(%SST(&REQUEST 21 8))
   CHGVAR VAR(&DBFID)    VALUE(%SST(&REQUEST 29 4))
/* IF FORMAT IS ZDAD0100 */
IF COND(&CSFMT *EQ 'ZDAD0100') THEN(DO)
   CHGVAR VAR(&DBFILE)   VALUE(%SST(&REQUEST 33 128))
   CHGVAR VAR(&DBLIB)    VALUE(%SST(&REQUEST 161 10))
   CHGVAR VAR(&DBMBR)    VALUE(%SST(&REQUEST 171 10))
   CHGVAR VAR(&DBAUT)    VALUE(%SST(&REQUEST 181 10))
   CHGVAR VAR(&DBBFIL)   VALUE(%SST(&REQUEST 191 128))
   CHGVAR VAR(&DBBLIB)   VALUE(%SST(&REQUEST 319 10))
   CHGVAR VAR(&DBOFIL)   VALUE(%SST(&REQUEST 329 10))
   CHGVAR VAR(&DBOLIB)   VALUE(%SST(&REQUEST 339 10))
   CHGVAR VAR(&DBOMBR)   VALUE(%SST(&REQUEST 349 10))
ENDDO


/* IF FORMAT IS ZDADO200 */
IF COND(&CSFMT *EQ 'ZDAD0200') THEN(DO)
  CHGVAR VAR(&DBNUM) VALUE(%SST(&REQUEST 33 4))
  CHGVAR VAR(&DBLIB2) VALUE(%SST(&REQUEST 37 10))
  ENDDO

/* IF FORMAT IS ZDAQ0100 */
IF COND(&CSFMT *EQ 'ZDAQ0100') THEN DO
   CHGVAR VAR(&DBSTMT)     VALUE(%SST(&REQUEST 33  18))
   CHGVAR VAR(&DBCRSR)     VALUE(%SST(&REQUEST 51  18))
   CHGVAR VAR(&DBSOPT)     VALUE(%SST(&REQUEST 69  2))
   CHGVAR VAR(&DBATTR)     VALUE(%SST(&REQUEST 71  2))
   CHGVAR VAR(&DBPKG)     VALUE(%SST(&REQUEST 73  10))
   CHGVAR VAR(&DBPLIB)     VALUE(%SST(&REQUEST 83  10))
   CHGVAR VAR(&DBDRDA)     VALUE(%SST(&REQUEST 93  2))
   CHGVAR VAR(&DBCMT)     VALUE(%SST(&REQUEST 95  1))
   CHGVAR VAR(&DBTEXT)     VALUE(%SST(&REQUEST 96  512))
ENDDO


/* IF FORMAT IS ZDAR0100 */
IF COND(&CSFMT *EQ 'ZDAR0100') THEN DO
   CHGVAR VAR(&DBLIBR)     VALUE(%SST(&REQUEST 33  20))
   CHGVAR VAR(&DBRDBN)     VALUE(%SST(&REQUEST 53  36))
   CHGVAR VAR(&DBPKGR)     VALUE(%SST(&REQUEST 69  2))
   CHGVAR VAR(&DBATTR)     VALUE(%SST(&REQUEST 89  20))
   CHGVAR VAR(&DBFULR)     VALUE(%SST(&REQUEST 109  256))
   CHGVAR VAR(&DBMBRR)     VALUE(%SST(&REQUEST 365  20))
   CHGVAR VAR(&DBFFMT)     VALUE(%SST(&REQUEST 385  20))
ENDDO


/* THE FOLLOWING PARAMETERS ADDITIONAL FOR FORMAT ZDAR0200 */
/* IF FORMAT IS ZDAR0200 */
IF COND(&CSFMT *EQ 'ZDAR0200') THEN DO
   CHGVAR VAR(&DBPLIB)     VALUE(%SST(&REQUEST 33  10))
   CHGVAR VAR(&DBPTBL)     VALUE(%SST(&REQUEST 43  128))
```

```
   CHGVAR VAR(&DBFLIB)     VALUE(%SST(&REQUEST 171  10))
   CHGVAR VAR(&DBFTBL)     VALUE(%SST(&REQUEST 181  128))
ENDDO


/* REMOTE COMMAND SERVER */
   CHGVAR VAR(&RCFMT)     VALUE(%SST(&REQUEST 21  8))
   CHGVAR VAR(&RCFID)     VALUE(%SST(&REQUEST 29  4))
   CHGVAR VAR(&RCPGM)     VALUE(%SST(&REQUEST 33  10))
   CHGVAR VAR(&RCLIB)     VALUE(%SST(&REQUEST 43  10))
   CHGVAR VAR(&RCNUM)     VALUE(%SST(&REQUEST 33  10))
   CHGVAR VAR(&RCDATA)     VALUE(%SST(&REQUEST 57  6000))

/* SIGNON SERVER DECLARES */
   CHGVAR VAR(&SOFNT)     VALUE(%SST(&REQUEST 21  8))
   CHGVAR VAR(&SOFID)     VALUE(%SST(&REQUEST 29 4))


/*********************************/
/*                               */
/* BEGIN MAIN PROGRAM            */
/*                               */


 CHGVAR VAR(&STATUS) VALUE('1') /* INITIALIZE RETURN +
                           VALUE TO ACCEPT THE REQUEST */

 /* ADD LOGIC COMMON TO ALL SERVERS */

 /* PROCESS BASED ON SERVER ID */
 IF COND(&APPLIC *EQ '*VPRT') THEN(GOTO CMDLBL(VPRT))   /* IF VIRTUAL PRINTER */
 IF COND(&APPLIC *EQ '*TFRFCL') THEN(GOTO CMDLBL(TFR))  /* IF TRANSFER FUNCTIO*/
 IF COND(&APPLIC *EQ '*FILESRV') THEN(GOTO CMDLBL(FLR)) /* IF FILE SERVERS */
 IF COND(&APPLIC *EQ '*MSGFCL') THEN(GOTO CMDLBL(MSG))  /* IF MESSAGING FUNCT */
 IF COND(&APPLIC *EQ '*DQSRV') THEN(GOTO CMDLBL(DATAQ)) /* IF DATA QUEUES */
 IF COND(&APPLIC *EQ '*RQSRV') THEN(GOTO CMDLBL(RSQL))  /* IF REMOTE SQL */
 IF COND(&APPLIC *EQ '*SQL') THEN(GOTO CMDLBL(SQLINIT)) /* IF SQL */
 IF COND(&APPLIC *EQ '*NDB') THEN(GOTO CMDLBL(NDB))     /* IF NATIVE DATABASE */
 IF COND(&APPLIC *EQ '*SQLSRV') THEN(GOTO CMDLBL(SQLSRV)) /* IF SQL */
 IF COND(&APPLIC *EQ '*RTVOBJINF') THEN(GOTO CMDLBL(RTVOBJ)) /* IF RETRIEVE OB*/
 IF COND(&APPLIC *EQ '*DATAQSRV') THEN(GOTO CMDLBL(ODATAQ))  /* IF D*/
 IF COND(&APPLIC *EQ '*QNPSERVR') THEN(GOTO CMDLBL(NETPRT))  /* IF NETWORK PRI*/
 IF COND(&APPLIC *EQ '*CNTRLSRV') THEN(GOTO CMDLBL(CENTRAL)) /* IF CENTRAL SER*/
 IF COND(&APPLIC *EQ '*RMTSRV') THEN(GOTO CMDLBL(RMTCMD))    /* IF RMTCMD/DPC */
 IF COND(&APPLIC *EQ '*SIGNON') THEN(GOTO CMDLBL(SIGNON))  /* IF SIGNON */

 GOTO EXIT

/* * * * * * * * * * * * * * * * * * * * * */
/* SUBROUTINES                             */
/*                                         */
/* * * * * * * * * * * * * * * * * * * * * */

/* VIRTUAL PRlNTER */
 VPRT:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT
/* TRANSFER FUNCTION */
 TFR:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT
```

```
/* FILE SERVERS */
  FLR:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT
/* MESSAGING FUNCTION */
  MSG:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT
/* DATA QUEUES */
  DATAQ:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT
/* REMOTE SQL */
  RSQL:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT
/* DATABASE INIT */
  SQLINIT:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT

/* NATIVE DATABASE */
       NDB:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT
/* DATABASE SQL */
  SQLSRV:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT

/* RETRIEVE OBJECT INFORMATION */
  RTVOBJ:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT

/* DATA QUEUE SERVER */
  ODATAQ:

   /* SPECIFIC LOGIC GOFS HERE */

   GOTO EXIT
/* NETWORK PRINT SERVER */
  NETPRT:

   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT
 /* CENTRAL SERVER */
  CENTRAL:
```

```
   /* SPECIFIC LOGIC GOES HERE */

   GOTO EXIT
 /* REMOTE COMMAND/DISTRIBUTED PROGRAM CALL */
  RMTCMD:

 /* IN THIS CASE IF A USER ATTEMPTS TO DO A REMOTE COMMAND/DISTRIBUTED  */
 /* PROGRAM CALL AND HAS A USERID OF userid THEY WILL NOT BE ALLOWED TO */
 /* CONTINUE.                                                           */
 IF COND(&USER *EQ 'userid') THEN(CHGVAR VAR(&STATUS) VALUE('0'))

      GOTO EXIT
 /* SIGNON SERVER */
  SIGNON:

  /* SPECIFIC LOGIC GOES HERE */

  GOTO EXIT

 EXIT:
ENDPGM
```

Appendix A. Parameter Formats for Exit Programs

This appendix provides the data structure for the second parameter to the exit point formats for each of the OS/400 servers. See “Chapter 5. Using Exit Programs” on page 41 for information about how to use these structures.

Exit Program Parameters

This section provides the exit program parameters.

File Server

The file server has one exit point defined:

QIBM_QPWFS_FILE_SERV Format PWFS0100

The QIBM_QPWFS_FILE_SERV exit point is defined to run an exit program for the following types of file server requests:

- Change file attributes
- Create stream file or create directory
- Delete file or delete directory
- List file attributes
- Move
- Open stream file
- Rename
- Allocate conversation

Note: For the file server, the exit program name is resolved when the QSERVER subsystem is activated. If you change the program name, you must end and restart the subsystem for the change to take effect.

Table 12 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QPWFS_FILE_SERV with the PWFS0100 format.

Table 12. Exit Point QIBM_QPWFS_FILE_SERV format PWFS0100

Offset				
Dec	**Hex**	**Type**	**Field**	**Description**
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For the file server, the value is *FILESRV.
20	14	BINARY(4)	Requested function	The function being performed: • **X’0000’** - Change file attributes request • **X’0001’** - Create stream file or directory request • **X’0002’** - Delete file or delete directory request • **X’0003’** - List file attributes request • **X’0004’** - Move request • **X’0005’** - Open stream file request • **X’0006’** - Rename request • **X’0007’** - Allocate conversation request

Table 12. Exit Point QIBM_QPWFS_FILE_SERV format PWFS0100 (continued)

Offset		Type	Field	Description
Dec	**Hex**			
24	18	CHAR(8)	Format name	The user exit format name being used. For QIBM_QPWFS_FILE_SERV, the format name is PWFS0100.
32	20	CHAR(4)	File access	If the requested function has a value of '5' (open), this field contains the following structure: • Read access, CHAR(1) **X'F1'** - Yes **X'F0'** - No • Write access, CHAR(1) **X'F1'** - Yes **X'F0'** - No • Read/Write access, CHAR(1) **X'F1'** - Yes **X'F0'** - No • Delete allowed, CHAR(1) **X'F1'** - Yes **X'F0'** - No
36	24	BINARY(4)	File name length	The length of the file name (the next field). The length can be a maximum of 16MB.
40	28	CHAR(*)	File name	The name of the file. The length of this field is specified by the File Name Length (the previous field). The file name is returned in the ISO/IEC 10646 (UCS—2 Level 1) character set, CCSID 61952.

Note:

- This format is defined by member EPWFSEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.
- For more information about the ISO/IEC 10646 (UCS—2 Level 1) character set, see *Information Standard, ISO/IEC 10646—1: Information technology — Universal—Octet Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane* , reference number ISO/IEC 10646—1: 1993(E).

  The APIs available to convert to and from UCS—2 Level 1 are iconv() and CDRCVRT.

Database Server

The database server has four different exit points defined:

1. QIBM_QZDA_INIT
   - Called at server initiation
2. QIBM_QZDA_NDB1
   - Called for native database requests
3. QIBM_QZDA_SQL1
   - Called for SQL requests
4. QIBM_QZDA_SQL2
   - Called for SQL requests
5. QIBM_QZDA_ROI1
   - Called for retrieving object information requests and SQL catalog functions

The exit points for native database and retrieving object information have two formats defined depending on the type of function requested.

The QIBM_QZDA_INIT exit point is defined to run an exit program at server initiation. If a program is defined for this exit point, it is called each time the database server is initiated.

Table 13 on page 61 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZDA_INIT using the ZDAI0100 format.

Table 13. Exit Point QIBM_QZDA_INIT format ZDAI0100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For this exit point, the value is *SQL.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QZDA_INIT the format name is ZDAI0100.
28	1C	BINARY(4)	Requested function	The function being performed The only valid value for this exit point is 0.
Note: This format is defined by member EZDAEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

The QIBM_QZDA_NDB1 exit point is defined to run an exit program for native database requests for the database server. Two formats are defined for this exit point. Format ZDAD0100 is used for the following functions:

- Create source physical file
- Create database file, based on existing file
- Add, clear, delete database file member
- Override database file
- Delete database file override
- Delete file

Format ZDAD0200 is used when a request is received to add libraries to the library list.

Table 14 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZDA_NDB1 with the ZDAD0100 format.

Table 14. Exit Point QIBM_QZDA_NDB1 format ZDAD0100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For this exit point, the value is *NDB.
20	14	CHAR(8)	Format name	The user exit format name being used For the following functions, the format name is ZDAD0100.

Table 14. Exit Point QIBM_QZDA_NDB1 format ZDAD0100 (continued)

Offset		Type	Field	Description
Dec	**Hex**			
28	1C	BINARY(4)	Requested function	The function being performed This field contains one of the following: • **X'1800'** - Create source physical file • **X'1801'** - Create database file • **X'1802'** - Add database file member • **X'1803'** - Clear database file member • **X'1804'** - Delete database file member • **X'1805'** - Override database file • **X'1806'** - Delete database file override • **X'1807'** - Create save file • **X'1808'** - Clear save file • **X'1809'** - Delete file
32	20	CHAR(128)	File name	Name of the file used for the requested function
160	A0	CHAR(10)	Library name	Name of the library that contains the file
170	AA	CHAR(10)	Member name	Name of the member to be added, cleared, or deleted
180	B4	CHAR(10)	Authority	Authority to the created file
190	BE	CHAR(128)	Based on file name	Name of the file to use when creating a file based on an existing file
318	13E	CHAR(10)	Based on library name	Name of the library containing the based on file
328	148	CHAR(10)	Override file name	Name of the file to be overridden
338	152	CHAR(10)	Override library name	Name of the library that contains the file to be overridden
348	15C	CHAR(10)	Override member name	Name of the member to be overridden
Note: This format is defined by member EZDAEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

Table 15 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZDA_NDB1 with the ZDAD0200 format.

Table 15. Exit Point QIBM_QZDA_NDB1 format ZDAD0200

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For this exit point, the value is *NDB.
20	14	CHAR(8)	Format name	The user exit format name being used. For the add to library list function, the format name is ZDAD0200.
28	1C	BINARY(4)	Requested function	The function being performed **X'180C'** - Add library list
32	20	BINARY(4)	Number of libraries	The number of libraries (the next field)
36	24	CHAR(10)	Library name	The library names for each library

Table 15. Exit Point QIBM_QZDA_NDB1 format ZDAD0200 (continued)

Offset		Type	Field	Description
Dec	Hex			
Note: This format is defined by member EZDAEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

The QIBM_QZDA_SQL1 exit point is defined to run an exit point for certain SQL requests that are received for the database server. Only one format is defined for this exit point. The following are the functions that cause the exit program to be called:

- Prepare
- Open
- Execute
- Connect
- Create package
- Clear package
- Delete package
- Stream fetch
- Execute immediate
- Prepare and describe
- Prepare and execute or prepare and open
- Open and fetch
- Execute or open

Table 16 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZDA_SQL1 with the ZDAQ0100 format.

Table 16. Exit Point QIBM_QZDA_SQL1 format ZDAQ0100

Offset		Type	Field	Description
Dec	Hex			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For this exit point, the value is *SQLSRV.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QZDA_SQL1, the format name is ZDAQ0100.

Table 16. Exit Point QIBM_QZDA_SQL1 format ZDAQ0100 (continued)

Offset		Type	Field	Description
Dec	**Hex**			
28	1C	BINARY(4)	Requested function	The function being performed This field contains one of the following: • **X'1800'** - Prepare • **X'1803'** - Prepare and describe • **X'1804'** - Open/Describe • **X'1805'** - Execute • **X'1806'** - Execute immediate • **X'1809'** - Connect • **X'180C'** - Stream fetch • **X'180D'** - Prepare and execute • **X'180E'** - Open and fetch • **X'180F'** - Create package • **X'1810'** - Clear package • **X'1811'** - Delete package • **X'1812'** - Execute or open
32	20	CHAR(18)	Statement name	Name of the statement used for the prepare or execute functions
50	32	CHAR(18)	Cursor name	Name of the cursor used for the open function
68	44	CHAR(2)	Prepare option	Option used for the prepare function
70	46	CHAR(2)	Open attributes	Option used for the open function
72	48	CHAR(10)	Extended dynamic package name	Name of the extended dynamic SQL package
82	52	CHAR(10)	Package library name	Name of the library for extended dynamic SQL package.
92	5C	BINARY(2)	DRDA indicator	• **0** - Connected to local RDB • **1** - Connected to remote RDB
94	5E	CHAR(1)	Commitment control level	• **'A'** - Commit *ALL • **'C'** - Commit *CHANGE • **'N'** - Commit *NONE • **'S'** - Commit *CS (cursor stability)
95	5F	CHAR(512)	First 512 bytes of the SQL statement text	First 512 bytes of the SQL statement
Note: This format is defined by member EZDAEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

Table 17 on page 65 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZDA_SQL2 with the ZDAQ0200 format.

The QIBM_QZDA_SQL2 exit point is defined to run an exit point for certain SQL requests that are received for the database server. The QIBM_QZDA_SQL2 exit point takes precedence over the QIBM_QZDA_SQL1 exit point. If a program is registered for the QIBM_QZDA_SQL2 exit point, it will be called and a program for the QIBM_QZDA_SQL1 exit point will not be called. The following are the functions that cause the exit program to be called:

- Prepare
- Open
- Execute
- Connect
- Create package
- Clear package
- Delete package
- Stream fetch
- Execute immediate
- Prepare and describe
- Prepare and execute or prepare and open
- Open and fetch
- Execute or open

Table 17 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZDA_SQL2 with the ZDAQ0200 format.

Table 17. Exit Point QIBM_QZDA_SQL2 format ZDAQ0200

0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For this exit point, the value is *SQLSRV.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QZDA_SQL1, the format name is ZDAQ0100.
28	1C	BINARY(4)	Requested function	The function being performed This field contains one of the following: • **X'1800'** - Prepare • **X'1803'** - Prepare and describe • **X'1804'** - Open/Describe • **X'1805'** - Execute • **X'1806'** - Execute immediate • **X'1809'** - Connect • **X'180C'** - Stream fetch • **X'180D'** - Prepare and execute • **X'180E'** - Open and fetch • **X'180F'** - Create package • **X'1810'** - Clear package • **X'1811'** - Delete package • **X'1812'** - Execute or open
32	20	CHAR(18)	Statement name	Name of the statement used for the prepare or execute functions
50	32	CHAR(18)	Cursor name	Name of the cursor used for the open function
68	44	CHAR(2)	Prepare option	Option used for the prepare function
70	46	CHAR(2)	Open attributes	Option used for the open function
72	48	CHAR(10)	Extended dynamic package name	Name of the extended dynamic SQL package
82	52	CHAR(10)	Package library name	Name of the library for extended dynamic SQL package.

Table 17. Exit Point QIBM_QZDA_SQL2 format ZDAQ0200 (continued)

92	5C	BINARY(2)	DRDA indicator	• **0** - Connected to local RDB • **1** - Connected to remote RDB
94	5E	CHAR(1)	Commitment control level	• **'A'** - Commit *ALL • **'C'** - Commit *CHANGE • **'N'** - Commit *NONE • **'S'** - Commit *CS (cursor stability)
95	5F	CHAR(10)	Default SQL collection	Name of the default SQL collection used by the AS/400 Database Server
105	69	CHAR(129)	Reserved	Reserved for future parameters
234	EA	BINARY(4)	SQL statement text length	Length of SQL statement text in the field that follows. The length can be a maximum of 32K.
238	EE	CHAR(*)	SQL statement text	Entire SQL statement
Note: This format is defined by member EZDAEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

The QIBM_QZDA_ROI1 exit point is defined to run an exit program for the requests that retrieve information about certain objects for the database server. It is also used for SQL catalog functions.

This exit point has two formats defined. These formats are described below.

Format ZDAR0100 is used for requests to retrieve information for the following objects:

- Library (or collection)
- File (or table)
- Field (or column)
- Index
- Relational database (or RDB)
- SQL package
- SQL package statement
- File member
- Record format
- Special columns

Format ZDAR0200 is used for requests to retrieve information for the following objects:

- Foreign keys
- Primary keys

Table 18 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZDA_ROI1 with the ZDAR0100 format.

Table 18. Exit Point QIBM_QZDA_ROI1 format ZDAR0100

Offset				
Dec	**Hex**	**Type**	**Field**	**Description**
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For the database server, the value is *RTVOBJINF.

Table 18. Exit Point QIBM_QZDA_ROI1 format ZDAR0100 (continued)

Offset		Type	Field	Description
Dec	**Hex**			
20	14	CHAR(8)	Format name	The user exit format name being used. For the following functions, the format name is ZDAR0100.
28	1C	BINARY(4)	Requested function	The function being performed This field contains one of the following: • **X'1800'** - Retrieve library information • **X'1801'** - Retrieve relational database information • **X'1802'** - Retrieve SQL package information • **X'1803'** - Retrieve SQL package statement • **X'1804'** - Retrieve file information • **X'1805'** - Retrieve file member information • **X'1806'** - Retrieve record format information • **X'1807'** - Retrieve field information • **X'1808'** - Retrieve index information • **X'180B'** - Retrieve special column information
32	20	CHAR(20)	Library name	The library or search pattern used when retrieving information about libraries, packages, package statements, files, members, record formats, fields, indexes, and special columns
52	34	CHAR(36)	Relational database name	The relational database name or search pattern used to retrieve RDB information
88	58	CHAR(20)	Package name	The package name or search pattern used to retrieve package or package statement information
108	6C	CHAR(256)	File name (SQL alias name)	The file name or search pattern used to retrieve file, member, record format, field, index, or special column information
364	16C	CHAR(20)	Member name	The member name or search pattern used to retrieve file member information
384	180	CHAR(20)	Format name	The format name or search pattern used to retrieve record format information
Note: This format is defined by member EZDAEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

Table 19 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZDA_ROI1 with the ZDAR0200 format.

Table 19. Exit Point QIBM_QZDA_ROI1 format ZDAR0200

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For the database server, the value is *RTVOBJINF.
20	14	CHAR(8)	Format name	The user exit format name being used. For the following functions, the format name is ZDAR0200.

Table 19. Exit Point QIBM_QZDA_ROI1 format ZDAR0200 (continued)

Offset		Type	Field	Description
Dec	**Hex**			
28	1C	BINARY(4)	Requested function	The function being performed This field contains one of the following: • **X'1809'** - Retrieve foreign key information • **X'180A'** - Retrieve primary key information
32	20	CHAR(10)	Primary key table library name	The name of the library that contains the primary key table used when retrieving primary and foreign key information
42	2A	CHAR(128)	Primary key table name (alias name)	The name of the table that contains the primary key used when retrieving primary or foreign key information
170	AA	CHAR(10)	Foreign key table library name	The name of the library that contains the foreign key table used when retrieving foreign key information
180	64	CHAR(128)	Foreign key table name (alias name)	The name of the table that contains the foreign key used when retrieving foreign key information
Note: This format is defined by member EZDAEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

Data Queue Server

The data queue server has one exit point defined:

QIBM_QZHQ_DATA_QUEUE format ZHQ00100

The exit point QIBM_QZHQ_DATA_QUEUE is defined to run an exit point program when the following data queue server requests are received:

- Query
- Receive
- Create
- Delete
- Send
- Clear
- Cancel
- Peek

Table 20 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZHQ_DATA_QUEUE with the ZHQ00100 format.

Table 20. Exit Point QIBM_QZHQ_DATA_QUEUE format ZHQ00100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For the data queue, server the value is *DATAQSRV.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QZHQ_DATA_QUEUE the format name is ZHQ00100.

Table 20. Exit Point QIBM_QZHQ_DATA_QUEUE format ZHQ00100 (continued)

Offset		Type	Field	Description
Dec	**Hex**			
28	1C	BINARY(4)	Requested function	The function being performed • **X'0001'** - Query the attributes of a data queue • **X'0002'** - Receive a message from a data queue • **X'0003'** - Create a data queue • **X'0004'** - Delete a data queue • **X'0005'** - Send a message to a data queue • **X'0006'** - Clear messages from a data queue • **X'0007'** - Cancel a pending receive request • **X'0012'** - Receive a message from a data queue without deleting it
32	20	CHAR(10)	Object name	Data queue name
42	2A	CHAR(10)	Library name	Data queue library
52	34	CHAR(2)	Relational operation	Relational operator for receive-by-key operation on the request **X'0000'** - No operator **'EQ'** - Equal **'NE'** - Not equal **'GE'** - Greater or equal **'GT'** - Greater than **'LE'** - Less or equal **'LT'** - Less than
54	36	BINARY(4)	Key length	Key length specified on the request
58	3A	CHAR(256)	Key value	Key value specified on the request
Note: This format is defined by member EZHQEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

Network Print Server

The network print server has two exit points defined:

1. QIBM_QNPS_ENTRY format ENTR0100
   - Called at server initiation
2. QIBM_QNPS_SPLF format SPLF0100
   - Called to process an existing spooled output file

The QIBM_QNPS_ENTRY exit point is defined to run an exit program when the network print server is started. The exit program can be used to verify access to the server. For more information, see *Printer Device Programming*, SC41-5713-03 .

Table 21 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QNPS_ENTRY with the ENTR0100 format.

Table 21. Exit Point QIBM_QNPS_ENTRY format ENTR0100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server

Table 21. Exit Point QIBM_QNPS_ENTRY format ENTR0100 (continued)

Offset		Type	Field	Description
Dec	**Hex**			
10	A	CHAR(10)	Server identifier	For the network print server, the value is QNPSERVR.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QNPS_ENTRY the format name is ENTR0100.
28	1C	BINARY(4)	Function identifier	The function being performed For QIBM_QNPS_ENTRY the value is X'0802'.
Note: This format is defined by member ENPSEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

The QIBM_QNPS_SPLF exit point is defined to run an exit program after the network print server receives a request to process an existing spooled output file. The program can be used to perform a function on the spooled file, such as fax the file. For more information, see *Printer Device Programming*, SC41-5713-03 .

Table 22 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QNPS_SPLF with the SPLF0100 format.

Table 22. Exit Point QIBM_QNPS_SPLF format SPLF0100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For the network print server the value is QNPSERVR
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QNPS_SPLF the format name is SPLF0100.
28	1C	BINARY(4)	Function identifier	The function being performed For QIBM_QNPS_SPLF, the value is X'010D'.
32	20	CHAR(10)	Job name	The name of the job that created the spooled file
42	2A	CHAR(10)	User name	The user profile of the job that created the spooled file
52	34	CHAR(6)	Job number	The number of the job that created the spooled file
58	3A	CHAR(10)	Spooled file name	The name of the spooled file being requested
68	44	BINARY(4)	Spooled file number	The number of the spooled file being requested
72	48	BINARY(4)	Length	Length of the spooled file exit program data
76	4C	CHAR(*)	Spooled file exit program data	Spooled file exit program data consists of additional information used by the exit program that has registered for exit point QIBM_QNPS_SPLF. The client application provides the spooled file exit program data.
Note: This format is defined by member ENPSEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

Central Server

The central server has three exit points defined:

1. QIBM_QZSC_LM format ZSCL0100
   - Called for license management requests
2. QIBM_QZSC_SM format ZSCS0100

- Called for system management requests

3. QIBM_QZSC_NLS format ZSCN0100
   - Called for conversion table requests

The QIBM_QZSC_LM exit point is defined to run an exit program for all license management requests received by the central server.

Table 23 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZSC_LM with the ZSCL0100 format.

Table 23. Exit Program QIBM_QZSC_LM format ZSCL0100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For the central server, the value is *CNTRLSRV.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QZSC_LM, the format name is ZSCL0100.
28	1C	BINARY(4)	Requested function	The function being performed This field contains one of the following: • **X'1001'** - Request license • **X'1002'** - Release license • **X'1003'** - Retrieve license information
32	20	CHAR(255)	Unique client name	The unique client name is used to identify a specific workstation across a network. The use of a licensed product is assigned to a workstation identified by the unique client name.
287	11F	CHAR(8)	License user handle	License user handle is used to ensure that the license requester and license releaser are the same. This value must be the same as when the license was requested.
295	127	CHAR(7)	Product identification	The identification of the product whose licensed use is requested
302	12E	CHAR(4)	Feature identification	The feature of the product
306	132	CHAR(6)	Release identification	The version, release, and modification level of the product or feature
312	138	BINARY(2)	Type of information	The type of information to be retrieved. The type of information field is only valid for the retrieve license information function This field contains one of the following: • **X'0000'** - Basic license information • **X'0001'** - Detailed license information
Note: This format is defined by member EZSCEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

The QIBM_QZSC_SM exit point is defined to run an exit program for all client management requests received by the central server.

Table 24 on page 72 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZSC_SM with the ZSCS0100 format.

Table 24. Exit Program QIBM_QZSC_SM format ZSCS0100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For the central server, the value is *CNTRLSRV.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QZSC_SM the format name is ZSCS0100.
28	1C	BINARY(4)	Requested function	The function being performed This field contains one of the following: • **X'1101'** - Set client active • **X'1102'** - Set client inactive
32	20	CHAR(255)	Unique client name	The client workstation name that is assigned to the licensed product
287	11F	CHAR(255)	Community name	The community name SNMP configuration field is used for authentication.
542	21E	CHAR(1)	Node type	The type of connection • **3** - Internet
543	21F	CHAR(255)	Node name	The name of the node For node type 3, the node name will be an Internet address.
Note: This format is defined by member EZSCEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

The QIBM_QZSC_NLS exit point is defined to run an exit program when the central server receives a request to retrieve a conversion map.

Table 25 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZSC_NLS with the ZSCN0100 format.

Table 25. Exit Program QIBM_QZSC_NLS format ZSCN0100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For the central server, the value is *CNTRLSRV.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QZSC_NLS, the format name is ZSCN0100.
28	1C	BINARY(4)	Requested function	The function being performed • **X'1201'** - Retrieve conversion map
32	20	BINARY(4)	From coded character set identifier (CCSID)	CCSID for existing data
36	24	BINARY(4)	To coded character set identifier (CCSID)	CCSID into which the data will be converted

Table 25. Exit Program QIBM_QZSC_NLS format ZSCN0100 (continued)

Offset		Type	Field	Description
Dec	**Hex**			
40	28	BINARY(2)	Type of conversion	Requested mapping type: • **X'0001'** - Round trip • **X'0002'** - Substitution mapping • **X'0003'** - Best-fit mapping
Note: This format is defined by member EZSCEP in files H, QRPGSRC, QRPGLESRC, QLBLSRC and QCBLLESRC in library QSYSINC.				

Remote Command and Distributed Program Call Server

The remote command/distributed program call server has one exit point defined:

QIBM_QZRC_RMT format CZRC0100

The QIBM_QZRC_RMT exit point is defined to call a program for either remote command or distributed program call requests.

The format of the parameter fields differ according to the type of request.

Table 26 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZRC_RMT with the CZRC0100 format when a remote command request is received.

Table 26. Remote Command requests for Exit Point QIBM_QZRC_RMT format CZRC0100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server
10	A	CHAR(10)	Server identifier	For the remote command server, the value is *RMTSRV.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QZRC_RMT, the format name is CZRC0100.
28	1C	BINARY(4)	Requested function	The function being performed **X'1002'** - Remote command
32	20	CHAR(10)	Reserved	Not used for remote command requests
42	2A	CHAR(10)	Reserved	Not used for remote command requests
52	34	BINARY(4)	Length of the next field	The length of the following command string
56	38	CHAR (6000)	Command string	Command string for remote command requests

Table 27 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZRC_RMT with the CZRC0100 format when a distributed program call request is received.

Table 27. Distributed Program Call requests for Exit Point QIBM_QZRC_RMT format CZRC0100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile that is calling the server

Table 27. Distributed Program Call requests for Exit Point QIBM_QZRC_RMT format CZRC0100 (continued)

Offset		Type	Field	Description
Dec	**Hex**			
10	A	CHAR(10)	Server identifier	For the distributed program call server, the value is *RMTSRV.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QZRC_RMT, the format name is CZRC0100.
28	1C	BINARY(4)	Requested function	The function being performed **X'1003'** - Distributed program call
32	20	CHAR(10)	Program name	Name of the program being called
42	2A	CHAR(10)	Library name	Library of the specified program
52	34	BINARY(4)	Number of parameters	The total number of parameters for the program call. This does not always indicate the number of parameters that follow.
56	38	CHAR(*)	Parameter information	Information about the parameters being passed to the specified program. All parameter strings have the following format regardless of the parameter usage type. The last field in the structure is specified for input/output parameter usage types. • BINARY(4) - Length of parameter information for this parameter • BINARY(4) - Maximum length of parameter • BINARY(2) - Parameter usage type – **1** - Input – **2** - Output – **3** - Input / output • CHAR(*) - Parameter string The maximum length of the parameter information is 6000 bytes. If the parameter information exceeds 6000 bytes, it is truncated.

Signon Server

The signon server, has one exit point defined:

QIBM_QZSO_SIGNONSRV format ZSOY0100

The exit point QIBM_QZSO_SIGNONSRV is defined to run an exit point program when the following signon server requests are received:

- Retrieve signon information
- Change password

Table 28 shows parameter fields and their descriptions for the exit program called at exit point QIBM_QZSO_SIGNONSRV with the ZSOY0100 format.

Table 28. Exit Point QIBM_QZSO_SIGNONSRV format ZSOY0100

Offset		Type	Field	Description
Dec	**Hex**			
0	0	CHAR(10)	User profile name	The name of the user profile associated with the request

Table 28. Exit Point QIBM_QZSO_SIGNONSRV format ZSOY0100 (continued)

Offset		Type	Field	Description
Dec	**Hex**			
10	A	CHAR(10)	Server identifier	For the signon server, the value is *SIGNON.
20	14	CHAR(8)	Format name	The user exit format name being used. For QIBM_QZSO_SIGNONSRV, the format name is ZSOY0100.
28	1C	BINARY(4)	Requested function	The function being performed • **X'7004'** - Retrieve signon information • **X'7005'** - Change password

Appendix B. Using the Database Server with DRDA

This appendix provides information about using Distributed Relational Database Architecture (DRDA) with the database server. Included are details on the rules, restrictions, and naming conventions used with DRDA.

SQL Packages

SQL packages bind SQL statements in an application program to a relational database. They are used to enhance the performance of applications that use dynamic SQL support by allowing the application to reuse information about the SQL requests. The database server is an application program that uses dynamic SQL requests. It supports the use of packages for frequently used SQL statements so that certain binding information can be reused.

SQL Package Names

The database server can be used as a gateway to other relational databases that use DRDA. The database server automatically creates one or more SQL packages on the target relational database. The package names are generated according to the attributes currently used by the server.

The package is created in a collection called QSQL400 on the application server if the relational database (RDB) is not an AS/400. If the RDB is an AS/400, the package is created in library QGPL. When the application server is not an AS/400, the package name is QZD**abcde**, in which **abcde** corresponds to specific parser options being used. Table 29 shows the options for the package name.

Table 29. Package name field options

Field	Field description	Options
a	Date format	ISO, JIS USA EUR JUL
b	Time format	JIS USA EUR, ISO
c	Commitment control/ decimal delimiter	*CS/period *CS/comma *CHG/period *CHG/comma *RR/period *RR/comma
d	String delimiter	apostrophe quote

Table 29. Package name field options (continued)

Field	Field description	Options
e	Maximum number of statements allowed for package	0 - 64 1 - 256 2 - 512 3 - 1024

When the application server is an AS/400, the package name is QZDA**abcdef**, in which **abcdef** corresponds to specific parser options being used.

Table 30. Package name field options

Field	Field description	Options
a	Date format	ISO, JIS USA EUR JUL MDY DMY YMD
b	Time format and naming convention	ISO, JIS and SQL naming USA and SQL naming EUR and SQL naming HMS and SQL naming ISO, JIS and system naming USA and system naming EUR and system naming HMS and system naming
c	Commit level and decimal point	*CS/period *CS/comma *ALL/period *ALL/comma *CHG/period *CHG/comma *NONE/period *NONE/comma
d	String delimiter	apostrophe quote

Table 30. Package name field options (continued)

Field	Field description	Options
e	Number of sections in package	0 - 64 1 - 256 2 - 512 3 - 1024
f	Date and Time separation	The high order bits of the character: **'1100'b** - One of the ISO formats for da **'1101'b** - Comma as date separation **'1110'b** - Period as date separation **'1111'b** - Colon as date separation The low order bits of the character: **'0001'b** - An ISO format of time **'0010'b** - Comma as time separator **'0011'b** - Period as time separator **'0100'b** - Slash as time separator **'0101'b** - Dash as time separator **'0110'b** - Blank as time separator

Cleaning Up SQL Packages

The packages used for DRDA functions are created automatically on your system as needed. You may want to periodically clean up these packages. To delete the packages, use the Delete SQL Package (DLTSQLPKG) command.

Delete the packages only if they are not used often. The package is created again if needed, but performance noticeably decreases when a package is created a second time.

Statement Naming Conventions

Table 31 provides a summary of the naming conventions enforced by the database server.

Table 31. Statement Naming Conventions

Statement	Dynamic SQL	Using an extended dynamic SQL package
Local	Statement name must adhere to AS/400 naming convention, although the format of STMTxxxx is suggested Cursor name must adhere to AS/400 naming conventions	Statement name must adhere to AS/400 naming convention, although the format of STMTxxxx is suggested Cursor name must adhere to AS/400 naming conventions

Table 31. Statement Naming Conventions (continued)

Statement	Dynamic SQL	Using an extended dynamic SQL package
DRDA	Statement name must be in the format of STMTxxxx Cursor name must be in the format: CRSRyyyy for non-scrollable cursors or SCRSRyyyy for scrollable cursors where yyyy is the same as xxxx.	Statement name must be in the format of Sxxxx Cursor name must be in the format of Cyy for non-scrollable cursors where yy is the same as xxxx and yy is between 1 and 15.

Notes:

1. The naming convention for statement names is not enforced on the local system, so a client application can share prepared statements with an AS/400 application using the QSQPRCED system API.
2. The server appends a blank to the beginning of any statement name in the format of STMTxxxx. A host application must then append a leading blank to share statements with client applications that use the format STMTxxxx. The server does not append a leading blank if the statement name is not in the format of STMTxxxx.

Rules and Restrictions When Using DRDA

When using the database server as a gateway to other RDBs using DRDA, some limitations in functions must be followed.

The following table shows the functions that have limitations when you are connected to a remote system from the database server.

Table 32. DRDA Functional Limits

Function	Limitation
Create package Clear package Delete package	Unsupported functions
Prepare	Enhanced prepare option not available when using DRDA.
Extended dynamic package support	• Only available when connected to an AS/400 running V2R3 or later • Can only access statements in a package using the naming convention of 'STMTxxxx' in which xxxx is the section number
Describe parameter markers	Only available when connected to an AS/400
Statement/cursor naming	See Table 31 on page 79 for statement naming conventions that are required when using DRDA.
Commit hold	Only valid if connected to an AS/400
Commit level *NONE	Not supported
Commit level *CHANGE	Only supported if the target RDB is an AS/400. All other RDBs require a *CS or *ALL commit level.

For more information on how these functions are used on the client, see *Client Access/400 for DOS and OS/2 API Reference*.

For more information on DRDA or database functions, see the following books:

- *Distributed Database Programming*, SC41-5702-01
- *Distributed Data Management*, SC41-5307-00
- *DB2 UDB for AS/400 SQL Reference*, SC41-5612-02
- *DB2 UDB for AS/400 SQL Programming*, SC41-5611-02

Appendix C. Using EZ-Setup, Operations Console, aand Operations Navigator with host servers

EZ-Setup, Operations Console, and Operations Navigator may connect to the signon, central, and remote command/distributed program call servers without a communication protocol running on the AS/400. That is, EZ-Setup and the Operations Console may connect before STRTCP or STRIPX has not been run. The path used permits EZ-Setup to perform some initial AS/400 setup before configuring or starting any communication protocols. This appendix describes how to determine if the communication path used by EZ-Setup and Operations Console is active and how to restart it if necessary.

For information on configuring the connection that is used by EZ-Setup consult the EZ-Setup online help. For information on configuring the connection that is used by Operations Console see *Configuring Operations Console*, SC41-5508-00.

The communication path used by EZ-Setup and Operations Console requires three jobs, QNEOSOEM, to be running in the QSYSWRK subsystem. The QSYSWRK subsystem has an auto start job for this communication path. The auto start job, QNEOSOEM, submits two other jobs with the name of QNEOSOEM in the QSYSWRK subsystem. If one of the jobs is not active, start it by issuing the following command:

```
QSYS/SBMJOB CMD(QSYS/CALL PGM(QSYS/QNEOSOEM)) JOB(QNEOSOEM)
JOBD(QSYS/QNEOJOBD) JOBQ(QSYS/QSYSNOMAX) PRTDEV(*JOBD) OUTQ(*JOBD)
USER(*JOBD) PRTTXT(*JOBD) SYSLIBL(*SYSVAL) INLLIBL(*JOBD)
LOGCLPGM(*YES) MSGQ(*NONE) SRTSEQ(*SYSVAL) LANGID(*SYSVAL)
CNTRYID(*SYSVAL) CCSID(*SYSVAL)
```

The command will start all three QNEOSOEM jobs if necessary.

Appendix D. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. 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 IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described 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:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation

Software Interoperability Coordinator
3605 Highway 52 N
Rochester, MN 55901-7829
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

Application System/400
AS/400
IBM
Operating System/400
OS/400
400

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

ActionMedia, LANDesk, MMX, Pentium, and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

Other company, product, and service names may be the trademarks or service marks of others.

Index

Q

R

S

U

Readers' Comments — We'd Like to Hear from You

AS/400e
Client Access Express Host Servers
Version 4

Publication No. SC41-5740-03

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

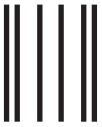
Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
SC41-5740-03

Cut or Fold
Along Line

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

IBM CORPORATION
ATTN DEPT 542 IDCLERK
3605 HWY 52 N
ROCHESTER MN 55901-7829

Fold and Tape **Please do not staple** Fold and Tape

Cut or Fold
Along Line

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC41-5740-03