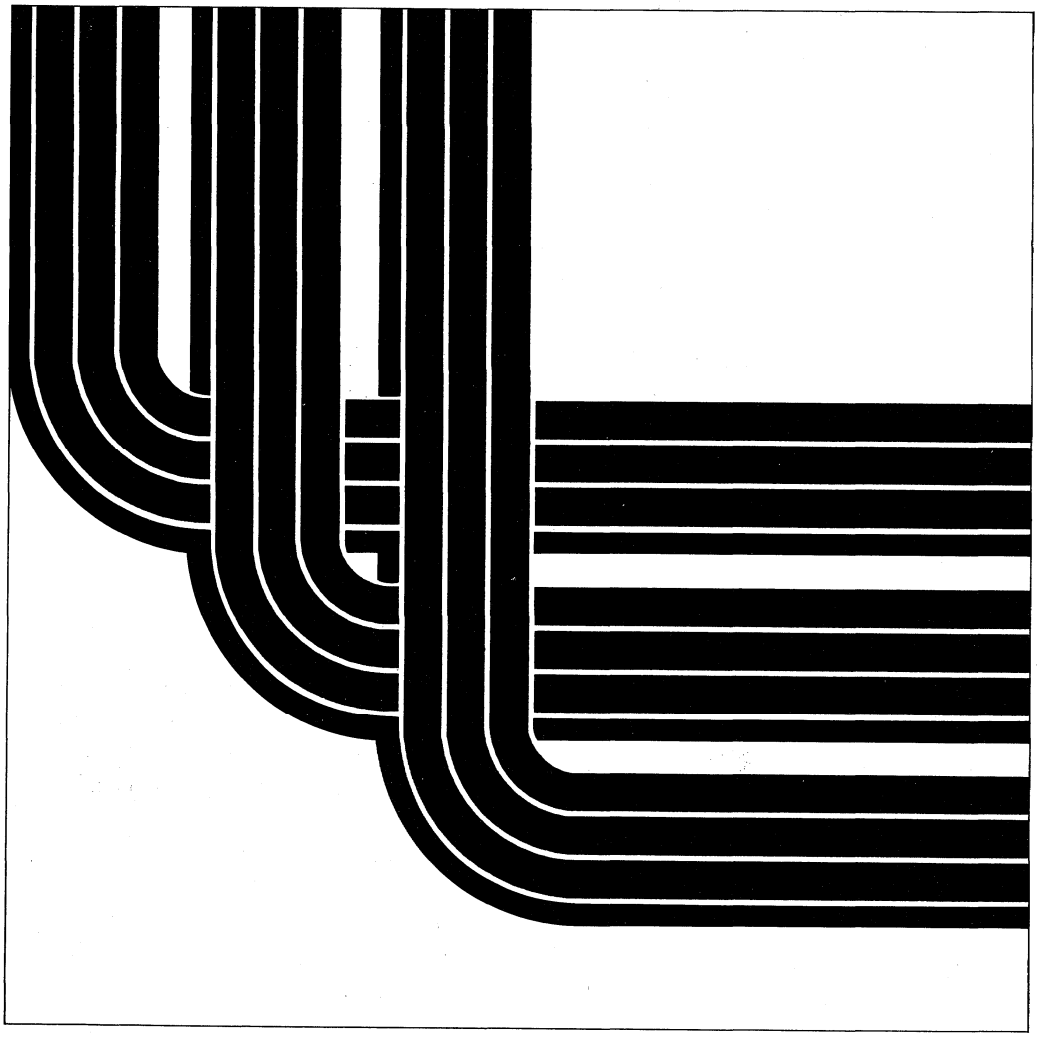
Application System/400

SC41-9600-00

Distributed Data Management Guide

Version 2



Application Development

Take Note!

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

First Edition (May 1991)

This edition applies to the licensed program IBM Operating System/400, (Program 5738-SS1), Version 2 Release 1 Modification 0, and to all subsequent releases and modifications until otherwise indicated in new editions. Make sure you are using the proper edition for the level of the product.

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

A form for readers' comments is provided at the back of this publication. If the form has been removed, you may address your comments to:

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

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

Contents

Figures

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates.

Any reference to an IBM licensed program or other IBM product in this publication is not intended to state or imply that only IBM's program or other product may be used.

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

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

Application System/400	AS/400	C/400
COBOL/400	FORTRAN/400	IBM
OfficeVision	Operating System/400	OS/400
RPG/400	SQL/400	System/370
400		

This publication could contain technical inaccuracies or typographical errors.

This manual may refer to products that are announced but are not yet available.

Information that has changed since Version 1 Release 3 Modification 0 is indicated by a vertical bar (|) to the left of the change.

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

Programming Interface

This data management guide is intended to help the customer use distributed data management (DDM) architecture on the AS/400 system. It primarily contains information about using DDM commands. The *DDM Guide* contains no programming interfaces for customers.

About This Guide

This guide contains Operating System/400 (OS/400) distributed data management (DDM) concepts, information about preparing for DDM communications, and DDM-related programming information.

Although this guide does contain some information about systems other than the AS/400 system, it does *not* contain all the information that the other system types may need to communicate with the AS/400 system using DDM. For complete information for a particular remote system type, refer to that system's documentation.

In this guide, the term *DDM* refers to the distributed data management architecture used by distributed data management (DDM) to define the protocols used for communicating between systems. DDM is also used to refer to the following:

- Terms used to discuss DDM architecture (for example, DDM jobs, conversations, functions, requests, and commands)
- Source and target implementations of the DDM architecture
- DDM files used by DDM to access remote files
- Non-AS/400 DDM products that support DDM (for example, System/36, System/38, and CICS/DDM)

Distributed relational database architecture also uses the DDM architecture. For more information about using distributed relational database architecture, see the *Distributed Database Guide*.

In this guide, the term *personal computer* refers to any of the IBM personal computers.

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

For a list of publications related to this guide, see the "Bibliography."

Who Should Use This Guide

This guide is intended for the application programmers and programmers who are using OS/400 distributed data management (DDM) to prepare a system to access data in remote files and to control access to local files by remote systems.

You should know general communications concepts, which are covered in *System Concepts*, GC41-9802. In addition, specific communications topics are discussed in the online index search. For more information on basic communications, see also the Discover/ IBM AS/400 course in the communications module. The Discover/ IBM AS/400 course can be ordered separately.

Chapter 1. Introduction to OS/400 DDM

This chapter describes the purpose of distributed data management (DDM), the functions that DDM supplies on the Application System/400* (AS/400*) system, and the concepts of Operating System/400* (OS/400*) DDM.

DDM is part of the Operating System/400 licensed program. OS/400 DDM as a source supports Level 2.0 and below of the DDM architecture. OS/400 DDM as a target supports Level 2.0 and below for **record file** (a file on disk in which the data is read and written in records) types and Level 3.0 and below of the DDM architecture for stream files (documents) and directories (folders). For additional information on the differences between the DDM architecture levels, see the *DDM Architecture: Reference* manual.

The DDM support on the AS/400 system allows application programs or users to access data files that reside on remote systems, and also allows remote systems to access data files on the local AS/400 system, as shown in Figure 1-1. Any system that supports the DDM architecture as a source system can access data (if authorized to do so) on any other system to which it is attached. The attached system must support DDM as a **target system** (the system that receives a request from another system to use one or more files located on the system).

However, the source and target systems must support compatible subsets and levels of the DDM architecture. (See "System Compatibility" on page 1-3.)

The folder management services (FMS) support allows personal computer users to access folders and documents that reside on an AS/400 target system. Remote systems that support Level 3.0 or Level 2.0 of the DDM architecture for the stream access method can access folders and documents on the local AS/400 system.

DDM extends the file accessing capabilities of the AS/400 database management support. In this manual, **database management** refers to the system function that controls **local** file processing; that is, it controls access to data in files stored on the local AS/400 system, and it controls the transfer of that data to requesting programs on the same system.

Distributed data management (DDM) controls **remote** file processing. DDM enables application programs running on one AS/400 system to access data files stored on another system supporting DDM. Similarly, other systems that have DDM can access files in the database of the local AS/400 system. DDM makes it easier to distribute file processing between two or more systems.

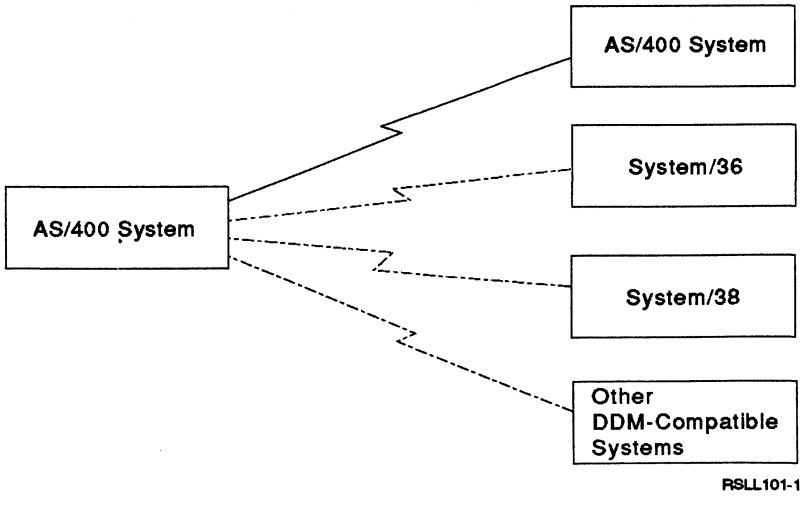


Figure 1-1. Source and Target Systems

Systems that use DDM communicate with each other using the advanced program-to-program communications (APPC) support or advanced peer-to-peer networking (APPN) support. See the *Communications Management Guide* and the *APPC Programmer's Guide* for information needed to use APPC and APPN.

Folder management services (FMS) allows local access to documents or folders that are on the AS/400 system. Personal computers may access folder management functions on the AS/400 system via DDM.

Note: Distributed data management for the IBM* Personal Computer uses the AS/400 portion of the PC Support/400 licensed program. See the *PC Support/400 User's Guide for DOS* or the *PC Support/400 User's Guide for OS/2* for information on how to use PC Support/400 on the AS/400 system.

As shown in Figure 1-2, the system on which a user application issues a request involving a remote file is called a **source system**. The system that receives the request for one of its files is called the **target system**. A system can be both a source and target system for separate requests received at the same time.

Using DDM, an application program can get, add, change, and delete data records in a file that exists on a target system. It can also perform file-related operations, such as creating, deleting, renaming, or copying a file from the target system to the source system. For an overview of the functions that can be done using DDM, see "Overview of DDM Functions" on page 1-3.

When DDM is in use, neither the application program nor the program user needs to know if the file that is needed exists locally or on a remote system. DDM handles remote file processing in essentially the same way as local file processing is handled on the local system, and the application program normally does not receive any indication of where the requested file is located. (However, in error conditions, messages are returned to the user that indicate, when necessary, that a remote system was accessed.) Informational messages about the use of target system files are included in the source system's job log.

When DDM is to be used, only the application programmer needs to know where the file is located and, using control language (CL) commands outside of the high-level language (HLL) programs, he can control which file is used. However, the programmer may also choose to use specific recovery functions to handle certain communications failures; the HLL programs may need to be changed to include handling any such failure.

Therefore, AS/400 BASIC, COBOL/400,* RPG/400,* C/400,* AS/400 PL/I and FORTRAN/400* programs that are compiled to process database files on the local system may not need to be changed or recompiled for DDM to process those same files when they are moved to or exist on a remote system.

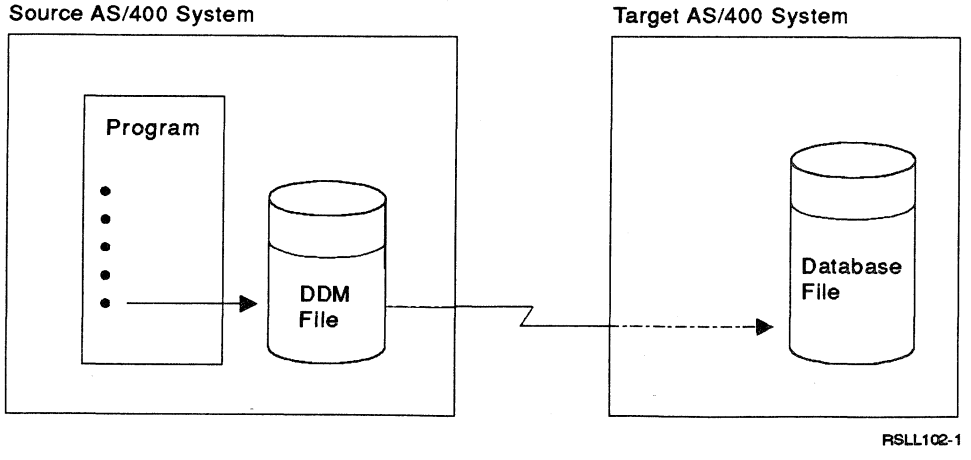


Figure 1-2. Source and Target Systems

For information about when programs on the source system need to be recompiled so they can access remote files as well as local files, see Chapter 3. For DDM limitations on the various languages, utilities, and applications, see Chapter 2.

System Compatibility

DDM can be used to communicate between systems that are architecturally different. For example, although the architectures of the AS/400 system and System/36 are different, these systems can use DDM to access files in each other's database. To successfully communicate with each other, each system must have an implementation of DDM that is compatible with Level 2.0 or below of the IBM DDM architecture. Also, each type of system may use all or only part of the IBM DDM architecture or may have extensions to the architecture.

If you are communicating with any non-AS/400 systems, you must consider the level of DDM support provided by those systems for such things as unique security considerations. OS/400 DDM security is discussed in Chapter 4.

| **Note:** The AS/400 system supports only advanced peer-to-peer networking (APPN) to allow DDM access to systems not directly linked to the local system. See Figure 1-5 on page 1-9 for an illustration of DDM access using APPN.

For a list of the DDM architecture manuals that supply the details about Level 3.0 or below of the IBM DDM architecture, see "Bibliography" on page H-1.

Overview of DDM Functions

This section gives an overview of the types of DDM functions that can be done on a target system.

The following *file* operations, normally specified in **HLL programs**, can be done on files at target systems:

- Allocating, opening, or closing one or more files
- Reading, writing, changing, or deleting records in a file

The following *file* and *nonfile* operations, normally specified in **CL programs** or by CL commands, can be done on files at the target systems:

- Copying the contents of a file.
- Performing operations on physical or logical file members (such as adding, clearing, or removing members), but only if the target is an AS/400 system or System/38.
- Accessing remote *files* for nondata purposes, such as:
  - Displaying information about one or more files, using commands such as Display File Description (DSPFD) and Display File Field Description (DSPFFD). These commands can display the file attributes of the DDM file on the source system or the file or field attributes of the remote file on the target system.
  - Controlling the locking of files on the target system, using the Allocate Object (ALCOBJ) and Deallocate Object (DLCOBJ) commands.
  - Deleting, renaming, creating, and changing files using the Delete File (DLTF), Rename Object (RNMOBJ), Create Physical File (CRTPF), Create Source Physical File (CRTSRCPF), Create Logical File (CRTLF), Change Physical File (CHGPF), Change Logical File (CHGLF), and Change Source Physical File (CHGSRCPF) commands.
- Accessing remote *systems* for nondata purposes:
  - Sending a CL command to the target system (an AS/400 system and a System/38 only) so it can be run there, instead of on the source system (where it may not be useful to run it), using the Submit Remote Command (SBMRMTCMD) command. The SBMRMTCMD command is the method you use to move, save, or restore files on a target system. For example, a Move Object (MOVOBJ) command might be sent to move a database file on the target system. (For typical uses of the SBMRMTCMD command, refer to its description in Chapter 5 or refer to the *CL Reference* manual for a more complete description.)

Various other *nonfile*-related operations, described later, can also be done on the target system.

Basic OS/400 DDM Concepts

The following topics give the basic concepts of OS/400 DDM:

- An overview of the three parts primarily used in DDM:
  - Source DDM
  - Target DDM
  - DDM file
- Example of DDM file use

Because remote file processing is much like local file processing, these topics should provide sufficient conceptual information for most users of DDM. Another section provides additional, more detailed concepts, and "Additional OS/400 DDM Concepts" on page 1-10 is intended primarily for the experienced programmer who wants or needs to know more about DDM.

From an end user's viewpoint, accessing data on a remote system is much the same as accessing data on the local system. The main difference is the additional time needed for the data link to pass the data between the systems whenever the remote file is accessed. Otherwise, the user or application program does not need to know whether the data being accessed came from a local or remote file. Refer to "Performance Considerations" on page 6-10 for additional considerations.

For DDM AS/400-to-AS/400 file processing, remote file processing is done much the same as local file processing. The purpose of this manual is to describe the things that are different for DDM. Also, because other systems can use DDM, those considerations and concepts are covered as needed to enable the AS/400 programmer to successfully prepare the system for using DDM.

The DDM concepts that are described on the following pages describe mainly AS/400-to-AS/400 remote file processing. For purposes of illustration, concepts that relate to System/36 and System/38 are shown in some examples. If you are using DDM on both System/36s and AS/400 systems, you should be aware that the concepts for both types are similar, except in the way they point to the remote file: An AS/400 system and a System/38 use a separate **DDM file** to refer to each remote file to be accessed; System/36 uses a network resource directory that contains one **network resource directory entry** for each remote file to be accessed.

Note: Although DDM supports other functions besides opening and accessing remote files, the concepts described in this chapter deal primarily with remote file accessing.

Parts of DDM

OS/400 DDM consists of three parts to handle remote file processing among the systems using DDM:

Source DDM (SDDM), the support on the source (or local) AS/400 system that is started, as needed, within a source job to do DDM functions. The SDDM translates requests for remote file access from source system application programs into DDM requests that are routed to the target system for processing. The SDDM support establishes and manages a DDM conversation with the target system that has the desired remote file.

Target DDM (TDDM), a target system job that is started on the target (or remote) system as a result of an incoming DDM request and that ends when the associated DDM conversation ends. The TDDM translates DDM requests for remote file access into data management requests on the target system and then handles the return of the information that is to be sent to the source system.

DDM file, a system object with type *FILE that exists on the source system to identify a remote file. It combines the characteristics of a device file and a database file. As a device file, the DDM file refers to a remote location name, local location name, device name, mode, and a remote network ID to identify a remote system as the target system. The DDM file appears to the application program as a database file and serves as the access device between a source system program and a remote file.

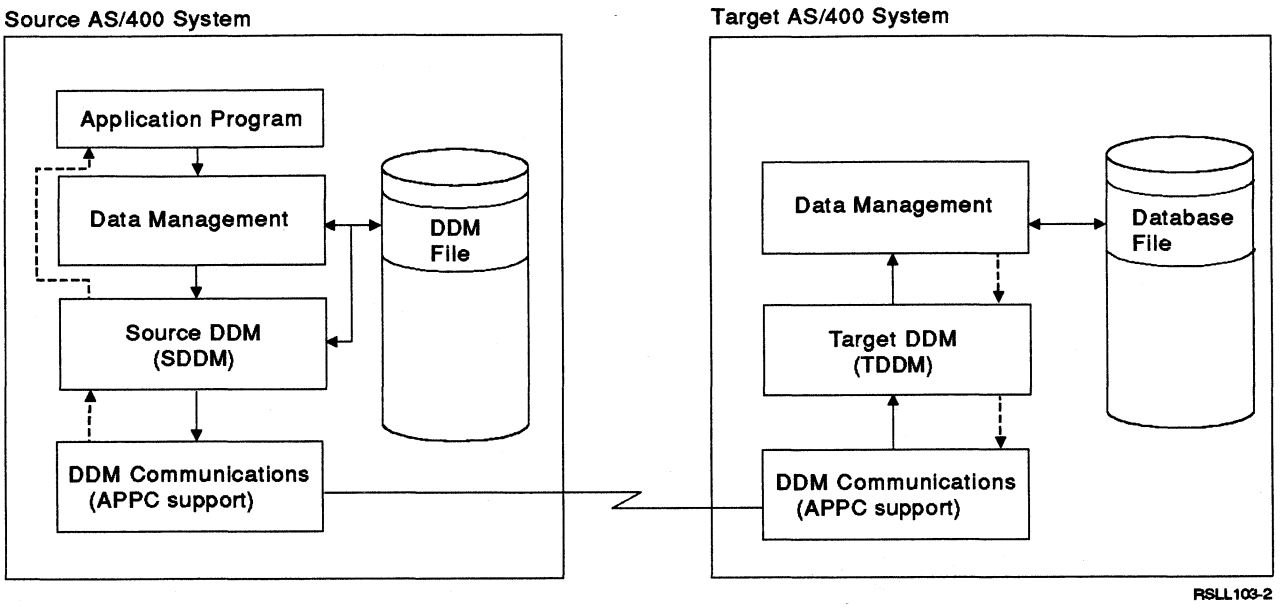


Figure 1-3. Communicating with DDM

Figure 1-3 shows how the basic parts involved in DDM communications on both systems relate to each other.

When a DDM file is accessed by a source system user or program, a **DDM conversation** is started between SDDM and TDDM for the job in which the program or user is operating.

Source DDM (SDDM)

When an application program first attempts to access a remote file, a search for the requested DDM file is done on the source system. As with local file processing, if the file name was not qualified with a library name, the current library list for the job in which the program is running is searched for the specified file. When the file is found, the system accesses the file, determines that it is a DDM file and starts the SDDM.

When the SDDM is started, it checks to see if a DDM conversation is already active between the source job starting the SDDM and the target system identified by the remote location and mode values in the DDM file. If a conversation exists that can be used, it is used. If not, a program start request is issued to the appropriate target system to start a TDDM (a target job) on the target system to establish a DDM. conversation between the SDDM and TDDM. Parameters that were automatically created from information in the DDM file about the remote file are passed when the remote system sends a program start request.

After the TDDM is started, the SDDM can forward each program request to the target job for processing. If, for example, input/output (I/O) operations are to be done on a remote file, the program opens the file and then issues the desired operation requests; the SDDM forwards the open request and the TDDM opens the remote file. Then the SDDM forwards each file operation request to the TDDM, and both of them handle the interchange of data between the application program and the remote file. When a DDM function is being processed, the requesting program waits for the function to be completed and the results to be received, the same as it does for local file operations.

For more detailed information about the SDDM on the AS/400 system, see "AS/400 System as the Source System" on page 1-10.

Target DDM (TDDM)

The TDDM is started when the remote system sends a program start request. The TDDM is started as a batch job on the target system. After the TDDM is started and a DDM conversation is established, the TDDM waits for a request (such as a file open or read operation, or a nonfile-related operation) to be sent by the SDDM.

When the TDDM receives a request to access an object on the target system, it searches for the requested object. If the object was not qualified with a library or path name, the current library list or current directory for the target job is searched.

When the requested object is found, the TDDM passes the first operation requested to database or folder management on the target system, which performs the operation on the object. When the operation is completed, database or folder management services returns the results of the operation to the TDDM, which passes it to the SDDM. The SDDM passes the results and any accompanying data (such as records requested on a read operation) to the application program. These actions are repeated for each subsequent I/O operation request received, until the object is closed. If an operation does not complete successfully, the SDDM returns an error message to the program, providing information about the error.

The TDDM and the target job remain active until the DDM conversation is ended by the source system job that started it. For more information about the TDDM on the AS/400 system, see "AS/400 System as the Target System" on page 1-13.

DDM File

A DDM file is a file on the source system that contains the information needed to access a data file on a target system. It is *not* a data file that can be accessed by a program for database operations. Instead, when a source system program specifies the name of a DDM file, the file information is used by DDM to locate the remote file whose data *is* to be accessed.

OS/400 DDM file information is based on *locations*. The remote location which is where the remote file is located, is specified using the remote location name (RMTLOCNAME) parameter on the Create DDM File (CRTDDMF) or Change DDM File (CHGDDMF) commands, and optionally, the remote network ID (RMTNETID). The local location is specified using the local location name (LCLLOCNAME). The device name (DEV) should only be specified if there is more than one APPC device associated with the same remote location and local location and you wish to select a specific device. The (MODE) parameter identifies the mode description that indicates the characteristics wanted for the session between the local and remote location.

Refer to the *APPC Programmer's Guide* for information on how these parameters are processed.

Note: In the System/38 environment, only the device name and mode can be specified. In this case, the remote location is taken from the device description.

The remote file name specified on the CRTDDMF or CHGDDMF commands must be in the format used by the remote system.

Another use of the DDM file is to submit control language (CL) commands to the target system to run on that system. In this case, the remote file normally associated with the DDM file is ignored. For more information on submitting commands, see "SBMRMTCMD (Submit Remote Command) Command" on page 5-2.

DDM File: Each OS/400 DDM file is created by the Create DDM File (CRTDDMF) command or by using the Work with DDM Files (WRKDDMF) command. The DDM file is stored as a file object in a library, the same as any other file or object. Each DDM file contains the following information:

DDM File Value and Description of Values

DDM file name
: The name of the DDM file on the source system that is used to identify a specific remote file.

Remote file name
: The actual file name of the remote file; that is, the name by which it is known on the target system. (For a target System/36, this is the file **label** of the remote file.)

Remote location name
: The name of the remote location where the remote file exists. This remote location name provides the data link to the target system (remote location) via APPN/APPC, over which a DDM conversation is established when this DDM file is accessed.

Device
: The name of the device on the source system used to communicate with the remote location.

Local location name
: The name of the local location. This is the name the target system knows your system by. Your system can consist of more than one local location.

Mode
: The name of the mode to be used to communicate between the local location and remote location.

Remote network ID
: The remote network ID to be used with the remote location. This value further qualifies the remote location name. Two locations with the same remote location name but different remote network IDs are viewed as two distinctly separate locations.

When a program refers to a DDM file, a DDM conversation with the target system is established. And, if the program is opening the DDM file to access records in the remote file, an open data path (ODP) to the remote file is also established.

Example Using the Basic Concepts: The following presents a sample application that uses DDM to access a remote file. The application could be run by a company that has warehouses located in several cities. Figure 1-4 on page 1-8 illustrates the relationships among the primary items included in a DDM file.

On an AS/400 system in Chicago, an Open Database File (OPNDBF) command requests that file CUST021 be opened for input. Because the file name was not qualified on the command, the library list for the source job is used to find the file, which is stored in the NYCLIB library.

Because CUST021 is a DDM file, the SDDM on the CHICAGO system is started in the source job when the file is opened. The SDDM uses the remote location and mode names (NEWYORK and MODENYC) from the DDM file to establish a DDM conversation with and start a target job (TDDM) on the appropriate target system (NEWYORK). The remote file to be accessed by the source system program is CUSTMAST in library XYZ.

The TDDM receives the remote file name from the SDDM and then allocates and opens the file named CUSTMAST, which corresponds to the DDM file named CUST021 on the source system.

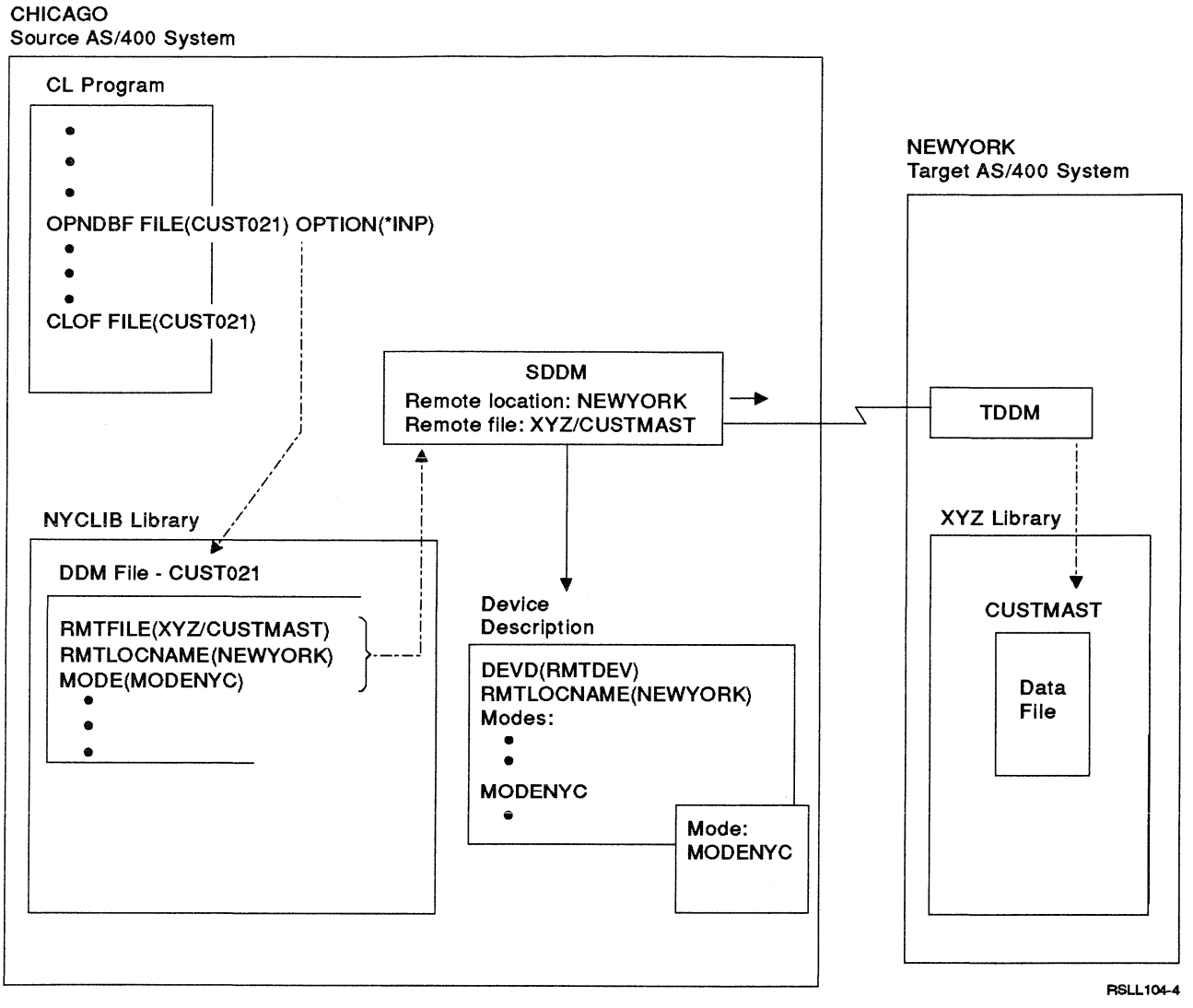


Figure 1-4. Relationships among DDM File Parameters and the Systems

The remote location name in the DDM file identifies the remote system where the file exists. The local system uses the remote location name as well as other values specified in the DDM file to select a device description. The device description can be either manually created or, if APPN is being used, automatically created and activated by the system. The SDDM establishes a DDM conversation with the target system using the values NEWYORK and MODENYC in the APPC remote location name. The APPC-related support must have been started on the target system before the request is issued by the SDDM. (No special support is required on the source system.)

Note: The APPN parameter on the Create Controller Description (APPC) (CRTCTLAPPC) and Create Controller Description (SNA Host) (CRTCTLHOST) commands determines whether or not the APPN support is used. Refer to the *APPC Programmer's Guide* for more information on using APPN, including how the system selects the device description to use.

Example Using the Basic Concepts in an APPN Network: As previously stated, the advanced peer-to-peer networking (APPN) support of an AS/400 system can be used to allow DDM access to systems not directly connected to the local system.

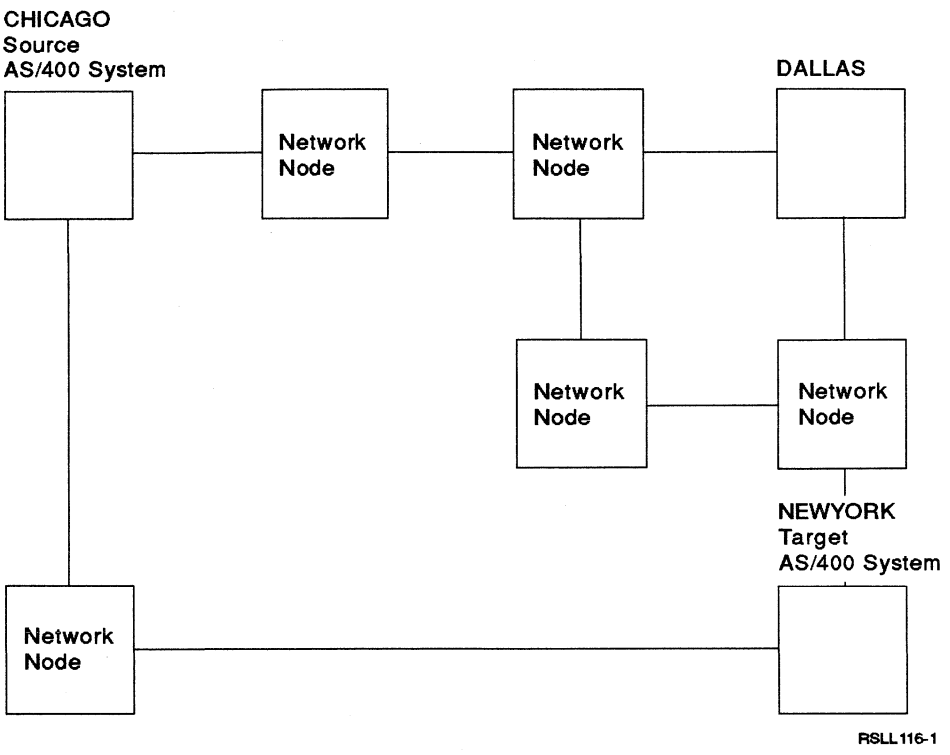


Figure 1-5. Using DDM in an APPN Network

Figure 1-4 shows a program on the Chicago system accessing a file on the New York system. Although the systems were shown as directly connected, the same DDM concepts apply if the network is configured as shown in Figure 1-5. When the DDM file CUST021 in Figure 1-5 is opened on the Chicago system, the APPN support finds the remote location named NEWYORK, determines the optimal path through the network, and establishes a DDM conversation with that location. Although there may be several other systems (network nodes) forwarding the data between CHICAGO and NEWYORK, the source DDM and target DDM function as if there were a direct connection between these two systems.

If the file CUSTMAST were moved from NEWYORK to some other system in the network (for example, DALLAS), then in this example, the DDM file at CHICAGO would need to be changed. The remote location name would be changed from NEWYORK to DALLAS. If a large number of systems in the network refer to the file CUSTMAST, then movement of the file results in a change to the DDM file at each of these systems. By using the AS/400 capability to have multiple local location names, maintenance of these files is reduced.

In Figure 1-5, the system NEWYORK could be given two local location names, NEWYORK and FILELOC. The DDM file at CHICAGO uses FILELOC as the remote location name. When access to file CUSTMAST is required, APPN finds the location FILELOC in the system named NEWYORK, and the DDM conversation is established as before.

If the file CUSTMAST is now moved from NEWYORK to DALLAS, the user at NEWYORK deletes the local location FILELOC from his system, and it is added to the system at DALLAS. This is done by using the APPN local location list. When the program in CHICAGO now attempts to access the file CUSTMAST, the APPN support finds the remote location FILELOC at the system in Dallas, and the DDM conversation is established to that system. The movement of CUSTMAST did not result in a change to the DDM file at CHICAGO.

This example shows the concept of multiple local locations and how reduced maintenance results when files are moved from one system to another. The example is not intended to suggest that a unique location name should be used for every file accessed through DDM. The decision of which files should be associated with separate

local locations should be based on such factors as the movement of these files and the number of remote systems accessing these files.

Additional OS/400 DDM Concepts

Most users of DDM will not need the information in the remainder of this chapter; it is intended primarily for experienced programmers who need to know more about DDM. These additional concepts should help a programmer understand and use the information in Chapter 5 and Chapter 6, which describe DDM-related command coding and working considerations.

Described are conceptual details and examples about:

- Program start requests, which start the TDDMs (target jobs)
- Open data paths (ODPs), used to access the files
- Remote location information
- DDM conversations, established for source and target communications
- Source and target jobs
- I/O operations within a job

AS/400 System as the Source System

When an application program or user in a source system job first refers to a DDM file, several actions occur as part of processing the request on the source system. All of these actions, as well as those required on the target system, must complete successfully before any operations (file or nonfile) requested by the source program can be done. When the DDM file is referred to:

- If the request is to open a file, its information is used simultaneously to create an open data path (ODP) on the source system and to start the SDDM support, which runs within the same job as the source program. The SDDM also uses the information: to convert the source system request into a DDM request, to communicate with the appropriate target system, and to establish a DDM conversation to be used for the source job. (The ODP is partially created with the DDM file information; it is not usable until the SDDM processes the remaining information after the DDM conversation is established.)
- The communications portion of DDM establishes a communications path with the target system using the advanced program-to-program communications (APPC) support. The target *system* is identified via the remote location information (remote location name, device, local location name, mode, remote network ID) specified in the DDM file, and the target *file* is identified by the remote file name. Using the remote location information, the TDDM is started on the target system and a DDM conversation is established when the remote system receives the program start request. The conversation is established the first time the remote file is accessed, but only if a conversation using the same remote location values for that target system does not already exist for the source job.
- After the DDM conversation is established, the SDDM (which can be used by multiple programs and multiple DDM files in the same source job) sends the DDM architecture command to the TDDM, for file-related requests. This command describes the file operation to be done and contains the name of the remote file (specified in the DDM file) to be accessed. (For nonfile-related requests, such as when the Submit Remote Command (SBMRMTCMD) command is used, the remote file name is *not* sent to the TDDM; the remote file name is ignored.)

The SDDM converts each program request for a file open or input/output operation (received via the DDM file and ODP) into an equivalent DDM command request and then sends it to the target system.

Figure 1-6 on page 1-11 shows the basic parts on the source AS/400 system that are involved in accessing remote files.

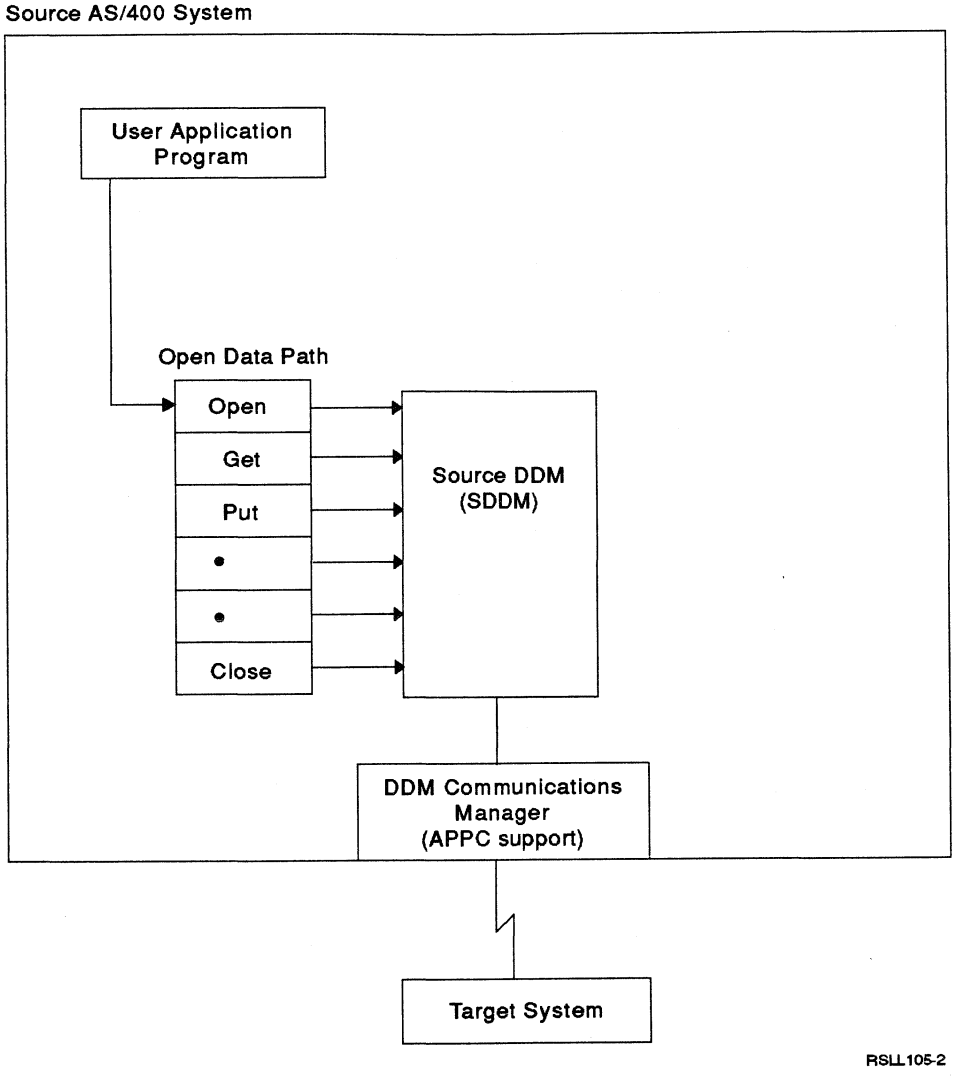


Figure 1-6. AS/400 System as the DDM Source System

After each request is handled by the target job, the DDM response from the target system is returned via APPC, converted by the SDDM into the appropriate form, and passed back to the user. The response may include data (if data was requested) or an indication of status (for other types of file access). The source program waits until the function completes and the results are received.

Figure 1-7 on page 1-12 shows a simplified example of the interchange of data between the source and target systems for a typical request to access a remote file.

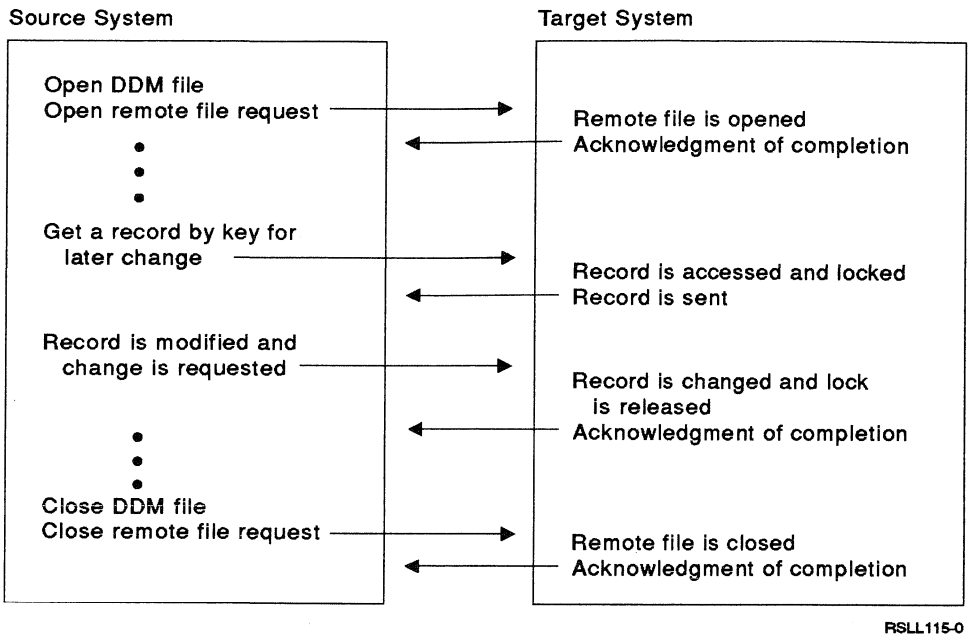


Figure 1-7. Typical Handling of an I/O Operation Request

After the first DDM file that was opened in the job is closed, the DDM conversation that it used is normally kept active. This allows the same program or another program in the job to use the same conversation when opening another DDM file, or doing other DDM-related operations. (For example, in Figure 1-9 on page 1-16, source job 3A has two DDM files using the same conversation.) This saves the time and resources required to establish a new conversation every time a new DDM file that uses the same remote location information is used in that job.

When a DDM file is closed, the DDM conversation remains active, but nothing happens in the conversation until the SDDM processes the next DDM-related request from a program. While it is not being used, however, the conversation can be dropped. This can occur if the DDMCNV job attribute's default value of *KEEP is changed to *DROP using the Change Job (CHGJOB) command, or if the Reclaim DDM Conversations (RCLDDMCNV) command or Reclaim Resources (RCLRSC) command is used while the job is active. The DDMCNV job attribute is described under "DDMCNV Parameter Considerations" on page 5-18 and all the commands are discussed in Chapter 5. Also, see "Controlling DDM Conversations" on page 6-8 for the conditions under which the conversation is considered unused.

Source System Actions Dependent on Type of Target System: If the target system is not another AS/400 system or System/38, only the DDM architecture commands defined in Level 2.0 and below of the DDM architecture are used. If the target is an AS/400 system or a System/38, then AS/400 system and System/38 extensions to the architecture are used to support some operations not defined by the Level 2.0 DDM architecture. Examples of System/38 and AS/400 extensions to the architecture are the Submit Remote Command (SBMRMTCMD) and processing file *members* of remote files. For further information, including restrictions on their use, see "SBMRMTCMD (Submit Remote Command) Command" on page 5-2. For creating a file when the source is an AS/400 system and the target is also an AS/400 system, an AS/400 extension is used.

Target systems that are not AS/400 systems or System/38s may not be capable of handling all of the functions that an AS/400 system or a System/38 can handle. For example, a System/36 does not support relative record processing and keyed record processing with one open; therefore, programs that mix accessing records in a file by key or relative record do not work if the file is on a System/36. In addition, target systems that do not support Level 2.0 of

the DDM architecture can only handle functions defined in the level they support.

Neither the System/36 nor the System/38 support access to folder management objects.

Note: An AS/400 system only allows access to folder management services (FMS) objects when the source supports Level 2.0 of the DDM architecture for **stream files** (files on disks in which data is read and written in consecutive fields without record boundaries) and directories, for example, the IBM Personal Computer using DDM. For more information on how to access folder management services from a personal computer, refer to the *PC Support/400 User's Guide for DOS* or *PC Support/400 User's Guide for OS/2*.

An AS/400 system as a source system does not support access to stream files and directories.

Language and utility specific restrictions are discussed in Chapter 2. For other possible restrictions, consult the specific target system documentation.

AS/400 System as the Target System

The AS/400 target DDM (or TDDM) is actually a job that runs a DDM-related target system program; it is started when the source system sends a program start request (an SDDM). For source AS/400 systems, the program start request is started on the source system using information contained in the IBM-supplied intersystem communications function (ICF) file for DDM. The remote location information in the DDM file being accessed is used to send the program start request to the appropriate target system.

The attributes of the target job are determined by the values specified on the Add Communications Entry (ADDCMNE) command, which is used on the target system to add a communications entry to the subsystem description used for the job. This command identifies the device description, the job description (including the library list for the target job), and the default user profile to be used by the subsystem.

For an PC Support/400 connection, the routing entry in the QIWS subsystem for DDM (CMPVAL ('DDM')), along with the device description the personal computer is connected to, is used to obtain the attributes of the target job.

After it is started, the TDDM does the following:

- For database files:
  - Handles communications with the source system via a DDM conversation established over an APPC data link or an PC Support/400 data link.
  - Converts the access requests from the source system into the equivalent AS/400 functions and runs them on the target system. Once the target object is located, the target system-created ODP and target database management services are used to access the object for whatever operation is requested. The TDDM can, for example, pass requests that open the object and then do requested I/O operations to the objects.
  - Includes AS/400 or System/38 extensions to the DDM Level 2.0 architecture for requests received from the source system (if the source is an AS/400 system or a System/38), which allow most AS/400 functions that operate on local systems to also work on remote AS/400 systems. For example, it might receive a SBMRMTCMD command from the source system (an AS/400 system or a System/38) to do a nonfile-related operation, such as using the CL command Replace Library List (RPLLIBL) to replace the library list within the current target job.
  - Converts target AS/400 responses to the equivalent DDM responses and sends them back to the source system. When the source system is an AS/400 system or System/38, the actual AS/400 or System/38 messages are sent back to the source system.
- For folder management services objects:

  Converts the DDM stream and directory access requests into the equivalent AS/400 folder management services functions and then runs them on the target system. The following commands are supported:

  Change Current Directory (CHGCD)
  Change File Attributes (CHGFAT)
  Close Directory (CLSDRC)

Close Document (CLOSE)
Copy File (CPYFIL)
Create Directory (CRTDRC)
Create Stream File (CRTSTRF)
Delete Directory (DELDRC)
Delete File (DELFIL)
Force Buffer (FRCBFF)
Get Data Stream (GETSTR)
Get Directory Entry (GETDRCEN)
List File Attributes (LSTFAT)
Load Stream File (LODSTRF)
Lock Data Stream (LCKSTR)
Open Directory (OPNDRC)
Open Document (OPEN)
Put Data Stream (PUTSTR)
Query Current Directory (QRYCD)
Query Space Available (QRYSPC)
Rename Directory (RNMDRC)
Rename File (RNMFIL)
Unload Stream File (ULDSTRF)
Unlock Data Stream (UNLSTR)

Figure 1-8 shows the basic parts on the target AS/400 system that are involved in processing the requested target file.

The TDDM runs as a separate batch job, just as any other user APPC target application or PC Support/400 target application. A new TDDM, using additional target system resources, is started for each distinct source system program start request received by the target system. There is one target job for each DDM conversation. Each TDDM can handle access requests for multiple files in the DDM conversation.

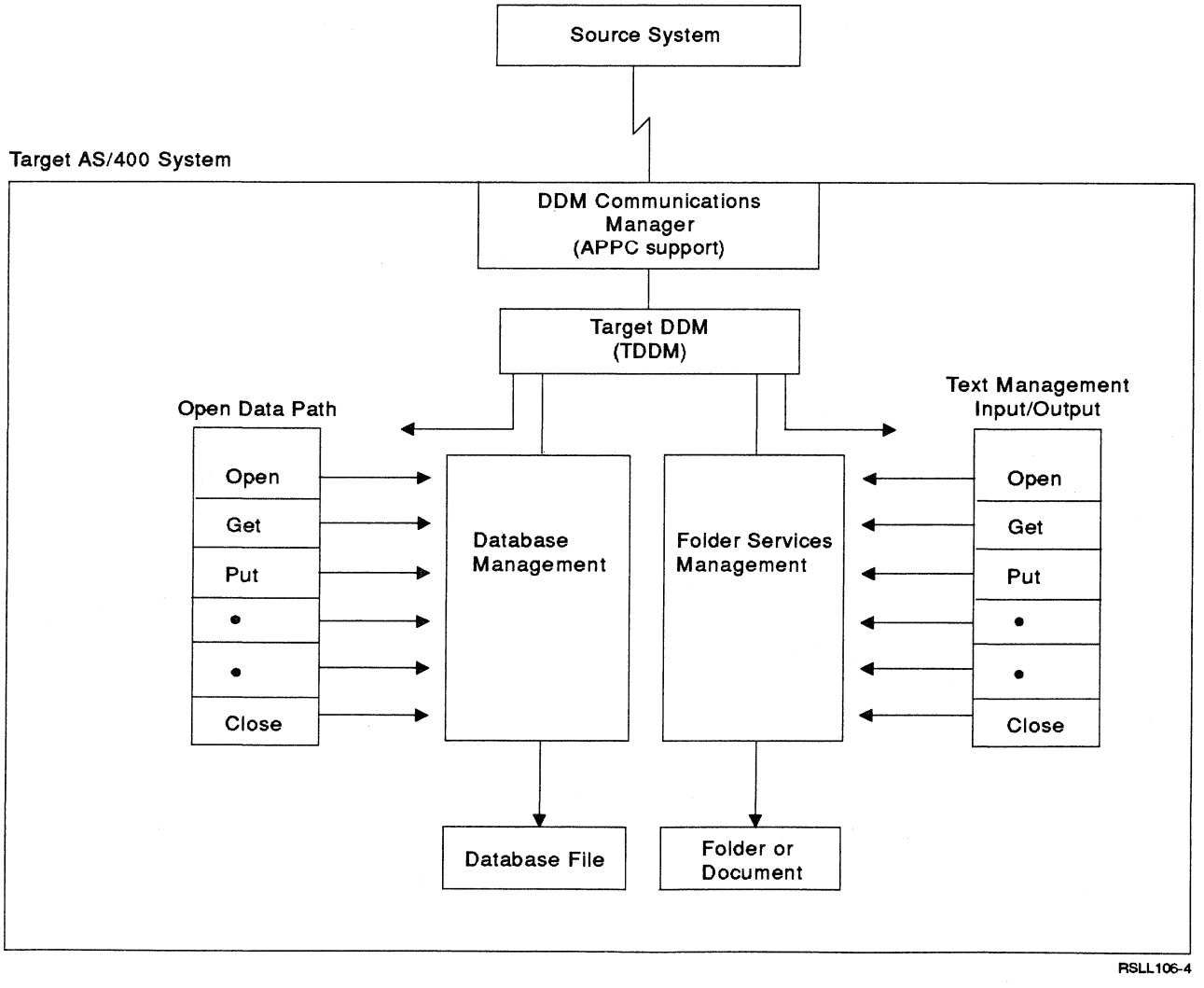


Figure 1-8. AS/400 System as the DDM Target System

The subsystem, user profiles, and system resources to be used by the TDDM are defined the same as they are for other types of jobs. For the APPC-related configuration information needed for DDM conversations, see the *OS/400* Communications Configuration Reference* or the *APPC Programmer's Guide*.

DDM-Related Jobs and DDM Conversations

This section provides additional information about source system jobs, target system jobs, and the DDM conversations used by those jobs.

For remote file processing, at least two separate jobs are used, one running on each system: a source job and a target job. (The source system job is the one in which the user application is running.) Multiple application programs can be running within a single source job. For each source job, there is a separate DDM conversation and target job for the remote location information specified in the DDM files used. If multiple DDM files are accessed within the same source job and each specifies the same remote location combination, they will share a DDM conversation. For each DDM conversation, there is one target job, which includes the TDDM.

The SDDM runs within a source job on the source system, and it can handle multiple DDM conversations with one or more target systems at the same time. And, for the same source job, one SDDM handles all the remote file access requests, regardless of how many target systems or remote files are involved. No separate job for the SDDM exists in the system.

If the source system DDM files involved all use the same remote location information to identify the target system, one TDDM job is created for each source system job that requests access to one or more files on the target system. If, for example, three programs in different jobs on the same source system (or three programs on different source systems) request access to one or more files on the same target system, the number of TDDMs (at least three, in this case) depends on the remote location information in the DDM files used to access that target system.

Figure 1-9 on page 1-16 shows three programs (one per source job) accessing six DDM files. The numbers in the upper set of boxes representing DDM files correspond to the same numbers in the lower set of boxes representing the associated remote files. These DDM files are using four different remote location descriptions to access six different remote files, all on the same target system. There are five DDM conversations because two source jobs (1A and 2A) are using the same remote location information, but a separate conversation is needed for each source job. Notice that source job 3A uses two DDM files that both use the same remote location information, so they share the same conversation and target job (5B).

Although Figure 1-9 on page 1-16 does not show any SDDMs or TDDMs, there is one for each job shown.

When the application program or the source job closes the DDM file on the source system, the DDM conversation and its associated target job ends, unless the following are true:

- The value of the DDMCNV attribute of the Change Job (CHGJOB) command for the source job is *KEEP (the system default).
- Any locks established during the job by the Allocate Object (ALCOBJ) command still exist.

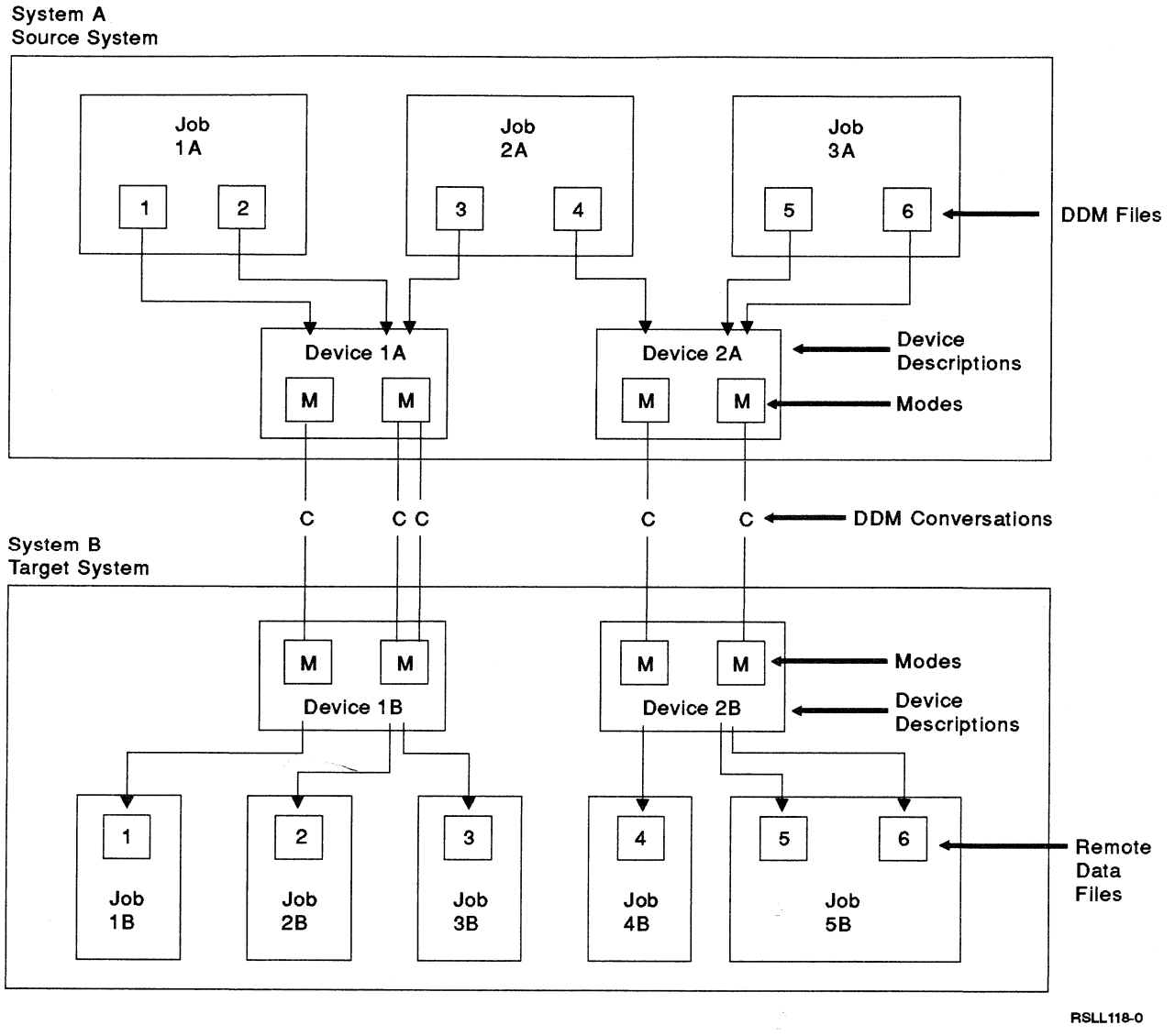


Figure 1-9. Relationships of DDM Source and Target Jobs

The CHGJOB and ALCOBJ commands are described in Chapter 5. If DDMCNV(*KEEP) is specified, the DDM conversation remains active and waits for another DDM request to be started.

From a performance viewpoint, if the DDM conversation is likely to be used again, *KEEP is the value that should be used. This saves the time and resources used on the target system to start each TDDM and establish the conversation and job.

Figure 1-10 on page 1-17 shows the relationship between the SDDM and two TDDMs on *different* target systems and Figure 1-11 on page 1-18 shows the relationship between the SDDM and two TDDMs on *one* target system.

An AS/400 system can be a source system and a target system at the same time, and two systems can be accessing files located on each other. Notice, however, that an AS/400 job can be a source job or a target job, but never both. A DDM file cannot refer to a remote file which is a DDM file; remote operations can only be done on remote files that are AS/400 or System/38 database files.

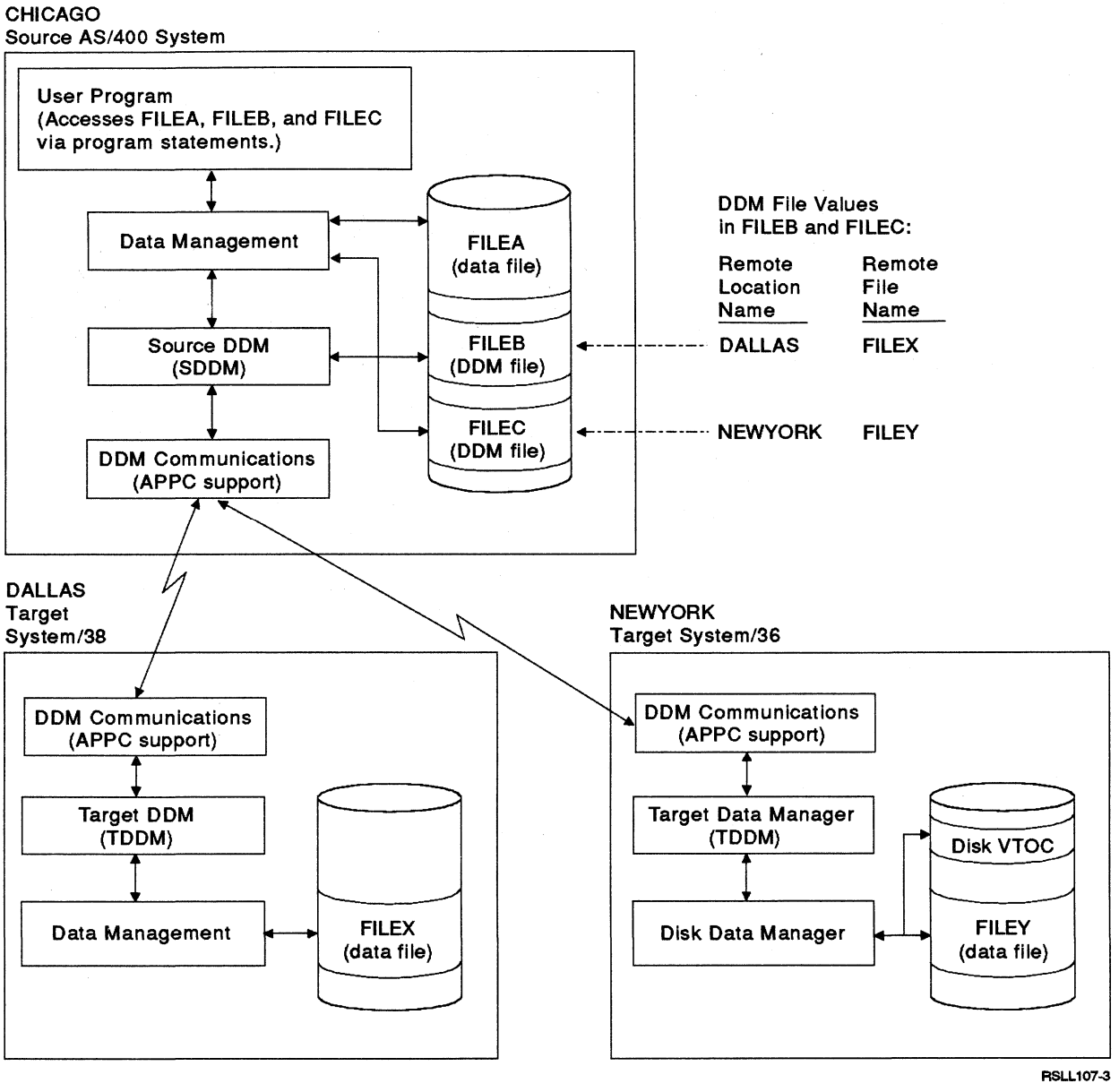


Figure 1-10. Example of Accessing Multiple Local and Remote Files. An AS/400 system with communications links to a System/38 and to a System/36.

Examples of Accessing Multiple Remote Files

Two examples follow that show a single application program using DDM to access multiple remote files. The first example shows the remote files on different target systems, and the second shows them on the same target system.

Example of Accessing Files on Multiple Systems

Figure 1-10 shows the relationships among the source system, its DDM files, and two target systems. One target system is a System/38 and the other is a System/36. Each system has DDM installed and uses an advanced program-to-program communications (APPC) support for communications.

The user program running on the source system is shown accessing three files: FILEA, FILEB, and FILEC. FILEA, located on the source system,

is accessed using only local data management. FILEB and FILEC are DDM files that correspond to remote files FILEX and FILEY (respectively) on different target systems. When the program opens FILEB and FILEC, DDM allows the program to access the corresponding remote files as if they were on the source system. Only the person who defines the DDM files needs to know where each file is located or what the file's name is on the remote system.

Example of Processing Multiple Requests for Remote Files

Figure 1-11 shows how multiple programs access multiple files on the same target system. This example shows a System/36 target system. The SDDM is shown handling requests for two files from two programs in different jobs, and two TDDMs are handling the requests on the target system (one TDDM for each requesting program). Notice that, although program B is accessing two files on the target system, only one TDDM is created if all the associated DDM files specify the same remote location information to identify the target system.

Notice that both programs A and B are sharing FILEA. However, because these programs are shown to be in separate jobs, they *cannot* share the same open data path (ODP) to FILEA. If they were in the same job, programs A and B could share both the ODP on the source system *and* the remote file. When multiple programs within the same job are accessing a remote file at the same time (via one TDDM for each program), the rules for file sharing are the same for remote files as for local files, and are based on how the SHARE parameter is specified on the Create DDM File (CRTDDMF), the Override with Database File (OVRDBF), and the Change DDM File (CHGDDMF) commands. See the *Data Management Guide* for information on share processing.

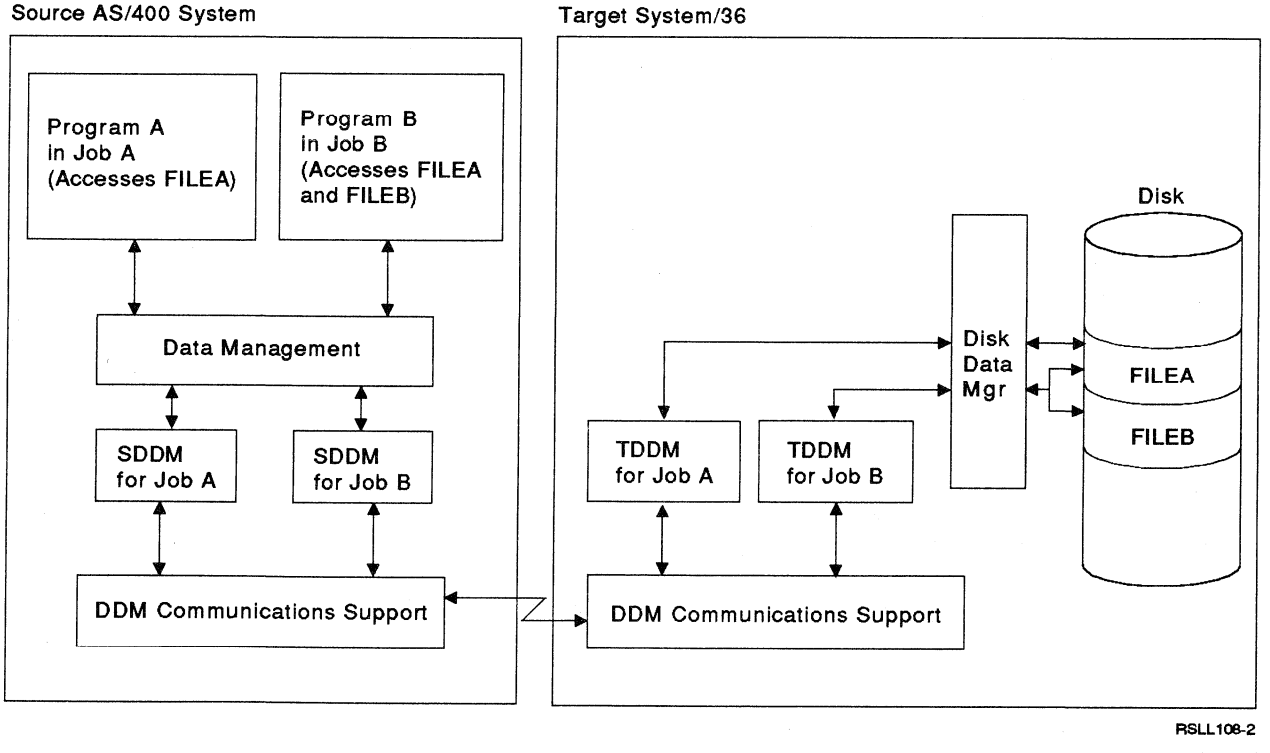


Figure 1-11. Example of Processing Multiple Program and File Requests

Chapter 2. Language, Utility, and Application Considerations for DDM

This chapter describes the language, utility, and application program support that is provided on the AS/400 system for DDM. This chapter indicates which languages, utilities, and application programs support DDM, and provides any DDM-specific information needed to properly access remote files. Language-specific information concerning access to Customer Information Control System for Virtual Storage (CICS) files is in Appendix E.

Programming Language Considerations

OS/400 DDM is supported by the following AS/400 languages:

- RPG/400 programming language
- COBOL/400 programming language
- AS/400 BASIC (interpretive and compiled forms)
- AS/400 PL/I
- Control Language (CL) (interactive and compiled forms)
- C/400 programming language
- FORTRAN/400 programming language

Note: AS/400 Pascal does not support DDM.

Considerations for All Languages

DDM files can be used as data files or source files by high-level language (HLL) programs. However, for CL, data description specifications (DDS), PL/I, and BASIC, if a DDM file is to be used as a source file, the target system must be an AS/400 system or a System/38, and the file referred to by the DDM file must be defined on the target AS/400 system or System/38 as a source file. That is, the remote file must have been created either by the Create Source Physical File (CRTSRCPF) command or as FILETYPE(*SRC) by the Create Physical File (CRTPF) command. These restrictions are not enforced by the RPG/400, COBOL/400, C/400, and FORTRAN/400 compilers, which allow source files to be used from both AS/400 and non-AS/400 target systems.

If a source file *member* name is specified when the target system is not an AS/400 system or a System/38, all the HLL compilers end compilation if the name of the source member specified on the SRCMBR parameter is different from the name of the DDM file specified on the SRCFILE parameter.

If programs that accessed local files are to access remote files, certain restrictions may require that a program be changed and recompiled. And, if the target system is not an AS/400 system or a System/38, externally described data must, in some cases, reside on the local (source) system. All of these restrictions are described under "Program Modification Requirements" on page 3-2.

If the target system is not an AS/400 system or a System/38, the number of records returned in the open feedback may not be valid.

If you do not specify a library name for the SRCFILE parameter, the first file found in the user's library list with the same name as the file you specified for the SRCFILE parameter is used as the source file.

HLL Program I/O Operations: The high-level language operations, shown in two parts in Figure 2-1 on page 2-2 and in Figure 2-2 on page 2-3, are supported by DDM for keyed or nonkeyed operations.

Figure 2-1. High-Level Language Operations Supported by DDM for Keyed or Nonkeyed Operations

	High-Level Languages			
OS/400 Database Operation	**RPG/400 Programming Language**	**COBOL/400 Programming Language**	**BASIC**	**PL/I**
Open file	OPEN	OPEN	OPEN	OPEN
Query file				
Read (keyed access)	CHAIN (key)	READ INVALID KEY	READ KEY	READ EQUAL
Read first/last[1]	*LOVAL *HIVAL	READ FIRST LAST	READ FIRST LAST	READ FIRST LAST
Read next	READ READE[2]	READ <NEXT> AT END	READ	READ NEXT
Read previous	READP	READ PRIOR AT END	READ PRIOR	READ PRV
Read next or previous[3] Next equal Previous equal Next unique Previous unique			READ = =, PRIOR	READ NXTEQL PRVEQL NXTUNQ PRVUNQ
Read (relative to start)[4]	CHAIN (rrn)	READ RELATIVE KEY	READ REC=	READ KEY
Release record lock	EXCPT or next I/O op	(next I/O op)	(next I/O op)	(next I/O op)
Force end of data	FEOD			
Position file[5]	SETGT SETLL	START KEY GREATER KEY NOT LESS KEY EQUAL	RESTORE	
Update record	UPDAT	REWRITE[6]	REWRITE	REWRITE
Write record	WRITE/ EXCPT	WRITE[6]	WRITE	WRITE
Delete record	DELET	DELETE[6]	DELETE	DELETE
Close file	CLOSE	CLOSE	CLOSE	CLOSE

Figure 2-2. High-Level Language Operations Supported by DDM for Keyed or Nonkeyed Operations

	High-Level Languages		
OS/400 Database Operation	**CL**	**C/400 Programming Language**	**FORTRAN/400 Programming Language**
Open file	OPNDBF	FOPEN, FREOPEN	OPEN
Query file	OPNQRYF		
Read (keyed access)			
Read first/last			
Read next	RCVF	FREAD, FGETC	READ
Read previous			BACKSPACE then READ
Read next or previous: Next equal Previous equal Next unique Previous unique			
Read (relative to start)			READ REC=
Release record lock		(next I/O op)	UNLOCK or next I/O op
Force end of data		FFLUSH	
Position file	POSDBF	FSEEK, FSETPOS	
Update record[7]		FWRITE, FPUTC, FFLUSH	WRITE REC=
Write record		FWRITE, FPUTC, FFLUSH	WRITE
Delete record			
Close file	CLOF	FCLOSE	CLOSE

Notes:

[1] For the RPG/400 language, if the keyed access path of a file specifies DESCENDING, then *LOVAL gets the last record in the file and *HIVAL gets the first record in the file.

[2] For duplicate keyed files, the RPG/400 language performs a READ NEXT operation and compares the key of the returned record to determine if the record qualifies. If so, the record is returned to the program; if not, an end-of-file indication is returned.

[3] If the remote file is on a non-AS/400 system, these operations cannot be performed using DDM.

[4] An AS/400 application program can open a *keyed* access open data path to a file and then access its records using both keyed and *relative record* access methods. Although OS/400 DDM supports the *combined-access* access method, a target system (such as System/36) may not. In this case, the AS/400 program can do relative record accessing of a keyed file on a non-AS/400 target system if the target system supports the *combined-by-record-number* access method and if the DDM file specifies that method. The combined-by-record-number access method is specified on an AS/400 system as ACCMTH(*ARRIVAL *BOTH) on the Create DDM File (CRTDDMF) command. If these values are not specified for the DDM file and the target system does not support the combined-access access method, relative record operations to a keyed file are rejected.

[5] Positioning operations (SETxx in the RPG/400 language, or START in the COBOL/400 language) do not return the record data to the application program. These operations also cause the file to be opened for random processing.

[6] COBOL/400 operations that change indexed or relative files can lock the record prior to the operation to make the record eligible. PL/I uses similar methods and options.

[7] For FORTRAN/400 programming language, access must be direct to update a record.

Commitment Control Support

AS/400 applications can commit or roll back transactions on remote AS/400 systems. However, DDM does not support the AS/400 journaling function. Before running applications, a user must create a journal on the target AS/400 systems for recoverable resources to be used under commitment control and then issue the Start Commitment Control (STRCMTCTL) command on the source system. The STRCMTCTL command does not support the Notify Object (NTFOBJ) command for DDM files.

Using DDM Files with Commitment Control: DDM files can be opened under commitment control. However, the following restrictions should be considered when working with these DDM files:

- If more than one DDM file is opened under commitment control, all opened files must have the same remote location name, local location name, device, mode, remote network ID, and transaction program name (TPN).
- If a DDM file and a remote SQL object are both running under commitment control, they must have the same remote location name, local location name, device, mode, remote network ID, and TPN.
- DDM files and local database files cannot be opened at the same time under commitment control.
- The Submit Remote Command (SBMRMTCMD) command should not be used to start or end commitment control.
- The target system specified from the AS/400 system working under commitment control must be another AS/400 system.

Note: If the communications line fails during a COMMIT operation, the source and target systems will do a ROLLBACK operation. However, the target system may successfully complete the COMMIT operation before the line fails, but the source system will always do a ROLLBACK operation.

Figure 2-3. High Level Language Commit and Rollback Commands

Operation	RPG/400 Programming Language	COBOL/400 Programming Language	PL/I	CL	C/400 Programming Language
Commit changes in transaction	COMMIT	COMMIT	PLICOMMIT	COMMIT	QXXCOMMIT
Cancel entire transaction	ROLBK	ROLLBACK	PLIROLLBACK	ROLLBACK	QXXROLLBCK

RPG/400 Considerations

RPG/400 programs and automatic report programs can both refer to DDM files. Generally, DDM file names can be specified in RPG/400 programming language anywhere a database file name can be specified, for both AS/400 and non-AS/400 target systems.

- DDM file names can be specified on the Create RPG Program (CRTRPGPGM) and Create Auto Report Program (CRTRPTPGM) commands:
  - To access remote files containing source statements, on an AS/400 system or a non-AS/400 system, a DDM file name can be specified on the SRCFILE parameter, and a member name can be specified on the SRCMBR parameter.
    - For AS/400 or System/38 target systems, a remote AS/400 or System/38 source file (and, optionally, member) can be accessed in the same manner as a local source file and member.
    - For non-AS/400 target systems, a remote source file can be accessed if both the PGM and SRCMBR parameter defaults are used on either command. Or, if a member name is specified, it must be the same as the DDM file name specified on the SRCFILE parameter. (The same is true for member names specified either on the /COPY statement of the input specifications used to create an automatic report program or as used by the compiler to include source specifications.)
  - To place the compiler listing in a database file on a target system, a DDM file name can be specified on the PRTFILE parameter of either command.
- A DDM file name and member name can be specified on the OUTFILE and OUTMBR parameters of the CRTRPTPGM command, but before the output produced by the command can be stored in the remote file referred to by the DDM file, the remote file must already exist. Also, as with local files, the record format of the remote file must match the required OUTFILE parameter format. Generally, this means that the target system must be an AS/400 system or a System/38.

When an RPG/400 program opens a DDM file on the source system, the following types of I/O operations can be performed on the remote file at the target system, for both AS/400 and non-AS/400 targets: CHAIN, CLOSE, DELET, EXCPT, FEOD, OPEN, READ, READE, READP, SETGT, SETLL, UPDAT, and WRITE.

Other considerations are:

- If the DDM file is declared in the program to be externally described, the RPG/400 compiler copies the external descriptions of the remote file referred to into the program at compile time. However, if the remote file is not on an AS/400 system or a System/38, the field declares for the record descriptions do not have meaningful names. Instead, all of the field names are declared as F*nnnnn* and the key fields are declared as K*nnnnn*.

  A recommended method for describing remote files, when the target is not an AS/400 system or a System/38, is to have the data description specifications (DDS) on the local system and enter a Create Physical File (CRTPF) command or a Create Logical File (CRTLF) command on the local system. Compile the program using the local file name. Ensure that the remote system's file has the corresponding field types and field lengths.

  To access the remote file, use the Override with Database File (OVRDBF) command preceding the program, for example:

  ```
  OVRDBF FILE(PGMFIL) TOFILE(DDMFIL) LVLCHK(*NO)
  ```

- A DDM file is also valid as the file specified in the RPG/400 program that will be used implicitly in the RPG/400 logic cycle.
- A record format name, if used, must match the DDM file name when the target system is not an AS/400 system or a System/38.
- An ADDROUT file created on a System/36 cannot be used on an AS/400 system. AS/400 System/36-Compatible RPG II uses 3-byte ADDROUT files, and RPG/400 programming language on an AS/400 system and System/38 uses 4-byte ADDROUT files.

COBOL/400 Considerations

COBOL/400 programs can refer to DDM files. Generally, DDM file names can be specified in COBOL/400 programming language anywhere a database file name can be specified, for both AS/400 and non-AS/400 target systems.

- DDM file names can be specified on the Create COBOL Program (CRTCBLPGM) command:
  - To access remote files containing source statements, on an AS/400 system or a non-AS/400 system, a DDM file name can be specified on the SRCFILE parameter, and a member name can be specified on the SRCMBR parameter.
    - For AS/400 or System/38 target systems, a remote AS/400 or System/38 source file (and, optionally, member) can be accessed in the same manner as a local source file and member.
    - For non-AS/400 target systems, a remote source file can be accessed if both the PGM and SRCMBR parameter defaults are used on the CRTCBLPGM command. Or, if a member name is specified, it must be the same as the DDM file name specified on the SRCFILE parameter.
  - To place the compiler listing in a database file on a target system, a DDM file name can be specified on the PRTFILE parameter of the CRTCBLPGM command.
- DDM file names can be specified as the input and output files for the COBOL/400 SORT and MERGE operation. (The work file for this operation cannot be a DDM file.)
- A DDM file can be used in the COBOL/400 COPY statement when the DDS option on that statement is used to copy one or all of the externally described record formats from the remote file referred to by the DDM file into the program being compiled. If this is done when the remote file is not on an AS/400 system or a System/38, the field declares for the record descriptions will not have meaningful names. Instead, all of the field names are declared as F*nnnnn* and the key fields are declared as K*nnnnn*.

  A recommended method for describing remote files, when the target is not an AS/400 system or a System/38, is to have the data description specifications (DDS) on the local system and enter a Create Physical File (CRTPF) command or a Create Logical File (CRTLF) command on the local system. Compile the program using the local file name. Ensure that the remote system's file has the corresponding field types and field lengths.

  To access the remote file, use the Override with Database File (OVRDBF) command preceding the program, for example:

  ```
  OVRDBF FILE(PGMFIL) TOFILE(DDMFIL) LVLCHK(*NO)
  ```

- DDM file names can be specified on a COPY statement:
  - If you do not specify the library name with the file name, the first file found with that file name in the user's library list is used as the include file.
  - If the target system is not an AS/400 system or a System/38, a DDM file name can be specified as the include file on a COPY statement, but the member name must be the same as the DDM file name.
- If the target system is a System/36, COBOL/400 programming language cannot be used to open a DDM file for output if the associated remote file has logical files built over it. For System/36 files with logical files, the open (for output) will fail because COBOL/400 programming language attempts to clear the file before using it.

When a COBOL/400 program opens a DDM file on the source system, the following statements can be used to perform I/O operations on the remote file at the target system, for both AS/400 and non-AS/400 targets: CLOSE, DELETE, OPEN, READ, REWRITE, START, and WRITE.

Direct File Support with COBOL/400: An AS/400 system does not support direct files as one of its file types. However, a COBOL/400 program on an AS/400 system can specify that a file be accessed as a *direct* file. (An AS/400 system normally creates direct files as *sequential* files.) A COBOL/400 program on an AS/400 system defines a file as a direct file by specifying RELATIVE on the SELECT statement. If the program is to open the file for output only (by

specifying OUTPUT on the OPEN statement), the file must be created with deleted records and contain no active records. This is also the file's condition when a non-AS/400 source system (such as System/36) uses DDM to create or clear the direct file on an AS/400 system, assuming that the file is created as described below.

An AS/400 system and System/38 support sequential and keyed file types. DDM recognizes sequential, keyed, and direct file types. For a non-AS/400 system to create a direct file on an AS/400 system using DDM, the DDM architecture command Create Direct File (CRTDIRF) is used.

When the CRTDIRF architecture command is issued from a non-AS/400 system to create the file, the file is created as a physical file and is designated as a direct file so that, for subsequent direct file access by non-AS/400 source systems, it will be identifiable to the other system as a direct file. If the file is not created in this way, an AS/400 system cannot later determine whether the file is a direct file or a sequential file, again, because an AS/400 system does not have direct files as one of its file types.

Therefore, if a COBOL/400 program on a system other than an AS/400 system or a System/38 needs to access an AS/400 or a System/38 file in a direct mode (that is, by relative record number) for output, the file must have been created by the CRTDIRF architecture command.

To support direct files on an AS/400 system for output only, the COBOL/400 OPEN statement clears and prepares a member of a file being opened. Therefore, existing AS/400 or System/38 files can be accessed via DDM files by COBOL/400 programs on other AS/400 systems or System/38s. For non-AS/400 target systems, relative files opened for output must be defined as direct files or an error occurs.

In summary:

- If a file is created on the local AS/400 system as a direct file by a program or user from a *non*-AS/400 system, the file can be accessed as a direct file by a COBOL/400 program from a remote non-AS/400 source system.
- If a file is created on the local AS/400 system by a program or user on the *same* AS/400 system, it cannot be accessed as a direct file by a non-AS/400 system because the AS/400 target system cannot determine, in this case, whether the file is a direct or sequential file.
- Any files created by a remote system can be used locally.

BASIC Considerations

Compiled BASIC programs and interpretive BASIC statements can refer to DDM files. In addition, DDM file names can be specified on the Create BASIC Program (CRTBASPGM), Start BASIC (STRBAS), and Execute BASIC Procedure (EXCBASPRC) commands.

- A DDM file name can be specified on the SRCFILE parameter, and a member name can be specified on the SRCMBR parameter of the CRTBASPGM, STRBAS, and EXCBASPRC commands, but only if the remote source file (and member) is on an AS/400 system or a System/38. If one of these commands refers to remote files on non-AS/400 or non-System/38 target systems, the operation fails.
- A DDM file can be used as the source file for the following BASIC commands in the BASIC session: FREE, LOAD, MERGE, PROC, REPLACE, SAVE, SRCFILE, and SUBPROC. It can also be used in the CHAIN BASIC statement.
- A DDM file name can be specified in the DECLARE FILE statement. The remote file that the DDM file refers to is used to bring in the field definitions for an externally described file. If this is done and the remote file is not on an AS/400 system or a System/38, the field declares for the record descriptions will not have meaningful names. Instead, all of the field names are declared as F*nnnnn* and the key fields are declared as K*nnnnn*.

  A recommended method for describing remote files, when the target is not an AS/400 system or a System/38, is to have the data description specifications (DDS) on the local system and enter a Create Physical File (CRTPF) command or a Create Logical File (CRTLF) command on the local system. Compile the program using the local file name. Ensure that the remote system's file has the corresponding field types and field lengths.

To access the remote file, use the Override with Database File (OVRDBF) command preceding the program, for example:

```
OVRDBF FILE(PGMFIL) TOFILE(DDMFIL) LVLCHK(*NO)
```

- A DDM file can be specified as the file used in the LISTFMT and LISTFMTP BASIC commands. These commands extract the file descriptions of the referred to remote file to list any fields used in the program.

When BASIC is used to open a DDM file on the source system the following statements can be used to perform I/O operations on the remote file at the target system, for both AS/400 and non-AS/400 targets: CLOSE, DELETE, INPUT, LINPUT, OPEN, READ, REREAD, RESTORE, REWRITE, and WRITE statements for processing record files, and GET and PUT statements for processing remote PL/I stream files.

PL/I Considerations

Compiled PL/I programs can refer to DDM files. In addition, DDM file names can be specified on the Create PL/I Program (CRTPLIPGM) command.

- A DDM file name can be specified on the SRCFILE parameter, and a member name can be specified on the SRCMBR parameter, but only if the remote source file is on an AS/400 system or a System/38. The same is true for specifying DDM file and member names on the %INCLUDE source directive statement. If the remote file referred to by the DDM file is not on an AS/400 system or a System/38, an error occurs if a DDM file name is specified on the CRTPLIPGM command or %INCLUDE statement.
- When a DDM file is accessed as the source file for a PL/I program, the margins used in the compilation of the PL/I source are the default values of 2 and 72. No other margin values can be specified.
- If a %INCLUDE DDS directive statement specifies the name of a DDM file, the record descriptions of the remote file are included in the compiled program. However, if the remote file is not on an AS/400 system or a System/38, the field declares for the record descriptions do not have meaningful names. Instead, all of the field names are declared as F*nnnnn* and the key fields are declared as K*nnnnn*.

A DDM file can be used to refer to remote record files or remote PL/I stream files. When a PL/I program opens a DDM file on the source system, the following types of statements can be used to perform I/O operations on the remote file at the target system, for both AS/400 and non-AS/400 targets: OPEN, CLOSE, READ, WRITE, REWRITE, and DELETE statements for processing record files, and GET and PUT statements for processing stream files.

Another consideration is if the target system is not an AS/400 system or a System/38, the POSITION parameter on a keyed READ statement to read from a remote file does not work if a value of NXTEQL, PRVEQL, NXTUNQ, or PRVUNQ is specified for the parameter. (The values of NEXT, PREVIOUS, FIRST, and LAST do work.) All the values are valid if the target system is an AS/400 system or a System/38.

CL Command Considerations

Both compiled CL programs and interactively entered CL commands can refer to DDM files. Generally, DDM file names can be specified in CL commands anywhere a database file name can be specified for both AS/400 and non-AS/400 target systems. But there are some limitations, and they are discussed later in this manual, primarily in Chapter 5.

Most of the information for using CL commands with DDM to access remote files is contained in Chapter 5 and Chapter 6.

Below are some examples of where DDM file names can be specified:

- DDM file names can be specified on many of the database file-related commands, such as the copy, display, and override file commands.
- DDM file names can be specified on the create file commands to access remote *source* files, but only if the target system is an AS/400 system or a System/38. A DDM file name can be specified on the SRCFILE parameter, and a member name can be specified on the SRCMBR parameter. If the remote source file referred to by the DDM file is not on an AS/400 system or a System/38, an error occurs. The considerations for remote AS/400 or System/38

source members are the same as for local source members.

- DDM file names can be specified on the FILE parameter of the Declare File (DCLF) command.

When a DDM file name is specified, some commands act on files on the source system, some act on target files, and some parameter values allow you to specify either a source or target file.

For summary charts that include the commands allowing DDM file names to be specified, see Appendix B.

C/400 Considerations

C/400 programs can refer to DDM files. Generally, DDM file names can be specified in C/400 programming language anywhere a database file name can be specified, for both AS/400 and non-AS/400 target systems.

Specify DDM file names on the Create C Program (CRTCPGM) command to do the following:

- Access remote files on an AS/400 or non-AS/400 system that contains source statements. To do this, specify a DDM file name on the SRCFILE parameter, and a member name on the SRCMBR parameter.

  Notes:

  1. For AS/400 or System/38 target systems, you access a remote AS/400 or System/38 source file (or member) in the same manner as a local source file and member.
  2. For non-AS/400 target systems, access a remote source file by using the same file name for the SRCMBR and the SRCFILE parameters.

- Place the compiler listing in a database file on a target system. To do this, specify a DDM file name on the PRTFILE parameter of the CRTCPGM command.

When using C/400 programming language, consider the following:

- If the target system is not an AS/400 system or a System/38, you can specify a DDM file name as the include file on the #INCLUDE source directive statement, but the member name must be the same as the DDM file name.
- C/400 programming language only supports sequential I/O operations.
- Although C/400 programming language does not directly support keyed files, key exceptions may occur if you are using a keyed file.

FORTRAN/400 Considerations

FORTRAN/400 programs can refer to DDM files. Generally, DDM file names can be specified anywhere a database file name can be specified, for both AS/400 and non-AS/400 target systems.

Specify DDM file names on the Create FORTRAN Program (CRTFTNPGM) command to do the following:

- Access remote files on an AS/400 or a System/38 target system that contains source statements. To do this, specify a DDM file name on the SRCFILE parameter, and a member name on the SRCMBR parameter.
- Place the compiler listing in a database file on a target system. To do this, specify a DDM file name on the PRTFILE parameter of the CRTFTNPGM command.

When using FORTRAN/400 programming language, consider the following:

- The ENDFILE statement cannot be used on a remote file.
- When a remote file is opened for direct access, records can be read from or written to the file.
- When a remote file is opened for sequential access and ACTION=WRITE, records can be written to the file. If the file is opened for sequential access and ACTION=READ or ACTION=READ/WRITE, records can be read from the file, but cannot be written to the file.

Utility Considerations

The following AS/400 utilities support DDM for accessing remote files:

- AS/400 System/38-compatible database tools:

  - System/38-compatible data file utility (DFU/38)
  - System/38-compatible query utility (Query/38)
- Data file utility for an AS/400 system (part of AS/400 Application Development Tools, Program 5728-PW1)
- Sort utility

Notes:

1. The following utilities do *not* support DDM: AS/400 Query, source entry utility (SEU), screen design aid (SDA), and advanced printer function utility.
2. Except when the System/38-compatible database tools or DFU/400 is being used, DDM does not support displaying lists of members in remote files. However, if the target system is an AS/400 system or a System/38, display station pass-through can be used to perform this function.

System/38-Compatible Database Tools

This section describes the System/38-compatible data file utility (DFU/38) and the System/38-compatible query utility (Query/38).

System/38-Compatible Data File Utility (DFU/38):

DFU/38 data entry applications can be created and used with DDM to work with remote files in the same manner as with local files. If a remote file is on an AS/400 system or System/38, most DFU/38 functions are performed with the remote file as though it is a local file. When creating or changing a DFU/38 application and the remote file is a logical file, the following consideration applies: either DDM files referring to each remote based-on file must exist on the source system, and the DDM file and library names must match those of the remote based-on files; or, alternatively, physical files with the same file and library names and the same record formats as the remote based-on files must exist on the source system. Because only the record formats are needed from the physical files, they need not contain data. Using this alternative, if the record formats of the remote based-on files are changed, the record formats on the source system must also be changed so that the record formats match.

However, DFU/38 does *not* support non-AS/400 or non-System/38 target systems. If you attempt to use DFU/38 with non-AS/400 or non-System/38 remote files, you may experience processing problems when trying to change or delete records in such a file. Although an AS/400 system does not prevent any user from creating and using such an application, the default field descriptions created on the source AS/400 system for the non-AS/400 or non-System/38 remote file would probably be too general to be useful. (These files appear to be physical files with one member, whose member name is the same as the file name. The file has one record format and within that format: one field for the entire record, if it is a nonkeyed file; two fields for keyed files, one for the key and one for the remainder of the record; or more than two fields for keyed files with separate key fields.)

All the DFU/38 commands can be used in applications that access local files or DDM files. And, wherever a local database file name can be specified on any of the DFU command parameters, a DDM file can also be specified, as long as any other limitations are met.

A DDM file name can be specified in the SRCFILE parameter of the Create DFU Application (CRTDFUAPP) or Retrieve DFU Source (RTVDFUSRC) command, but only if the target system is an AS/400 system or a System/38 and if the target file is a source physical file.

System/38-Compatible Query Utility (Query/38):

The System/38-compatible query utility (Query/38) can be used with DDM to create and use interactive or batch query applications. (DDM considerations with interactive database query are described in "OS/400 Database Query" on page 2-13.) If the target system is an AS/400 system or a System/38, most of these functions can be performed as though the remote file is a local file. When creating or changing a Query/38 application and the remote file is a logical file, the following consideration applies: either DDM files referring to each remote based-on file must exist on the source system, and the DDM file and library names must match those of the remote based-on files; or, alternatively, physical files with the same file and library names and the same record formats as the remote based-on files must exist on the source system. Because only the record formats

are needed from the physical files, they need not contain data. Using this alternative, if the record formats of the remote based-on files are changed, the record formats on the source system must also be changed so that the record formats match.

If the target system is not an AS/400 system or a System/38, you should refer to a local file for the format and fields that describe the data in the remote file, and then use the Override Database File (OVRDBF) command to override the local file with a DDM file when the Query/38 application is run. This is explained further in "Non-AS/400 or Non-System/38 Query/38 Example." The local file used to create (or re-create) the query must have the same record format name as the source description of the non-AS/400 or non-System/38 target file. The default record format name is the name of the source DDM file.

Although Query/38 can create an application that uses a file on a non-AS/400 or non-System/38 system, the default field descriptions created on the source AS/400 system for the non-AS/400 remote file probably would be too general to be useful. (These files appear to be physical files with one member, whose member name is the same as the file name. The file has one record format and within that format: one field for the entire record, if it is a nonkeyed file; two fields for keyed files, one for the key and one for the remainder of the record; or more than two fields for keyed files with separate key fields.)

Non-AS/400 or Non-System/38 Query/38 Example:

The following is an example of how to create a local file and use it to define the data that is to be queried in a non-AS/400 or non-System/38 remote file.

Assume that a DDM file named RMTS36FILE exists on your AS/400 system and it refers to a remote System/36 file that you want to query. You can perform the following steps to: determine the attributes of the remote System/36 file; locally create a physical file that has the attributes of the remote file; and define, create, and run the Query/38 against the remote file.

1. Use the Display File Field Description (DSPFFD) command and specify SYSTEM(*RMT) to display the attributes of the remote file associated with the RMTS36FILE DDM file.

   ```
   DSPFFD  FILE(RMTS36FILE) SYSTEM(*RMT)
   ```

   In this example, the displayed results would show that the remote file's record length is 80 characters, its record format name is RMTS36FILE, and it has two fields: K00001, with 12 characters (starting in position 1), and F00001, with 68 characters (starting in position 13). The K in field K00001 indicates it is the key field for this format.

2. Using the DDS and the above information before defining your Query/38 application, create a local physical file and call it LCLS36FILE. The DDS might look something like this:

   ```
   A          R RMTS36FILE
   A            CUSNO          6A
   A            BILLCODE       6A
   A            ADDR1         15A
   A            ADDR2         15A
   A            ADDR3         15A
   A            ZIP            5A
   A            AMTOWE         7S 2
   A            OUTBAL         7S 2
   A            MISC           4A
   A          K CUSNO
   A          K BILLCODE
   ```

   Three main rules must be followed when defining the local file:

   - The record format name must be the same as the record format name displayed by the Display File Field Description (DSPFFD) command.
   - Key integrity must be maintained. In this case, the key must be 12 characters long, and must start at the beginning of the file in position 1.
   - The total record length must be the same as the record length displayed by the DSPFFD command.

3. Define your Query/38 application using the local file created in step 2. Because the remote file is a non-AS/400 file, OPTIMIZE(*NO) should be specified on the query command. (See "Query/38 Optimization" on page 2-12 for more information.)

4. Before your Query/38 application is run, issue the following Override Database File (OVRDBF) command:

   ```
   OVRDBF  FILE(LCLS36FILE) TOFILE(RMTS36FILE)
   ```

When the Query/38 application is run, this command overrides the local file you created with the DDM file that is associated with the desired target file.

5. Run your Query/38 application using the Query Data (QRYDTA) command. The net effect is that a query of the remote file is done using the local file description.

Query/38 Output Considerations:

Query/38 output to an existing non-AS/400 or a non-System/38 target file is possible, but only under specific circumstances. Query/38 allows output to any local or remote file only if the file is sequential and if its field attributes match those attributes required by the Query/38 application. If both conditions are not met, Query/38 rejects the specified output file before the Query/38 application runs.

Because the source system description of a non-AS/400 or a non-System/38 target file is very general, its field attributes probably do not match the attributes required by the Query/38 application. Therefore, in most cases, Query/38 rejects that file if it is specified for output. It works, however, if the Query/38 output consists of one alphanumeric field only, and if the record length of the target file is large enough to hold this field.

Query/38 Command Considerations: All the Query/38 commands can be used in applications that access local files or DDM files. And, wherever a local database file name can be specified on any of the Query/38 command parameters, a DDM file can also be specified, as long as any other limitations are met.

Note: If a Query/38 command uses a DDM file associated with a remote file on a non-AS/400 or a non-System/38 target system, either the DDM file should specify LVLCHK(*NO) or an OVRDBF command should be used to override that parameter with *NO. This is recommended to avoid level-checking problems with the target file.

A DDM file name can be specified in the SRCFILE parameter of the Create Query Application (CRTQRYAPP) or Retrieve Query Source (RTVQRYSRC) command, but only if the target system is an AS/400 system or a System/38 and if the target file is a source physical file.

Query/38 Optimization: Query/38 has an optimization function, but because it causes OS/400 database query to be used, the feature cannot be used when the query is performed against a remote file that is not on an AS/400 system or a System/38. Because OS/400 database query does not exist on non-AS/400 systems or non-System/38s, the optimization function cannot be used by the source AS/400 system when performing a query against a non-AS/400 or a non-System/38 remote file. (See "OS/400 Database Query" on page 2-13.)

Therefore, when a Query/38 application is being created or changed that accesses a remote file on a non-AS/400 system or a non-System/38, the OPTIMIZE parameter on the Create Query Application (CRTQRYAPP), Create Query Definition (CRTQRYDEF), or Change Query Definition (CHGQRYDEF) command must be changed to *NO. Specifying OPTIMIZE(*NO) forces Query/38 to read the file sequentially, which can be done with non-AS/400 target files. If the default of *YES is used, an error occurs when the Query/38 application is run.

Similarly, if the Design Query Application (DSNQRYAPP) command is used to create and run queries that are to be performed on a non-AS/400 target file, the *Optimize Query* prompt on the Application Creation display must be changed from Y to N.

Existing Query/38 Application Considerations:

Existing Query/38 applications, if they are to query remote files, must be re-created in all cases, even if the target system is an AS/400 system or a System/38. If the target system is an AS/400 system or a System/38, the re-created application that uses a DDM file is defined and run as if the remote file is a local file. The optimization feature can be used to get the records from the target AS/400 system or the target System/38.

Data File Utility for AS/400 System

DFU data entry applications can be created and started with DDM to work with remote files in the same manner as with local files. Most DFU functions are performed with the remote file as though it were a local file. When creating or changing a DFU function of Application Develop-

ment Tools and the remote file is an AS/400 or System/38 logical file, the following consideration applies: either DDM files referring to each remote based-on file must exist on the source system, and the DDM file and library names must match those of the remote based-on files; or, alternatively, physical files with the same file and library names and the same record formats as the remote based-on files must exist on the source system. Because only the record formats are needed from the physical files, they need not contain data. Using this alternative, if the record formats of the remote based-on files are changed, the record formats on the source system must also be changed so that the record formats match. Similar considerations apply when the remote file is a System/36 logical file.

DFU supports AS/400, System/38, and System/36 remote files. However, DFU does not prevent you from using non-AS/400, non-System/38, or non-System/36 remote files and you may experience problems when using such files.

Non-AS/400 or System/36 files are program-described files. DFU allows you to use either a local or remote file containing RPG/400 file and input specifications to define these data files.

OS/400 Database Query

The database interactive query function, provided by OS/400, supports DDM files. This support is used by PC Support/400, OfficeVision/400*, and System/38-compatible query utility if OPTIMIZE(*YES) is specified. You can query remote files using the Open Query File (OPNQRYF) command, but only if the remote files are on a target AS/400 system or a target System/38. If multiple remote files are specified on one OPNQRYF command, they must all exist on the same target system and use the same remote location information. (See "System/38-Compatible Query Utility (Query/38)" on page 2-10 for more information on the System/38-compatible query utility support.)

If the target system is an AS/400 system or a System/38, a query request is created and sent to the target system via the DDM file that the query refers to. If the target system is not an AS/400 system or a System/38, the query request cannot be processed and an error message is created. However, the query utility on the System/38 can be used to query non-AS/400 remote files. (Again, see "System/38-Compatible Query Utility (Query/38)" on page 2-10 for details.)

If the target system is a System/38 and the source is an AS/400 system, or if the target system is an AS/400 system and the source is a System/38, OPNQRYF or PC Support/400 cannot use group-by and join functions. An error results.

Multiple Remote Files: Database query allows accessing of either multiple local files or multiple remote files (via DDM files) at the same time, but not both. If all the files are remote, they must all reside on the same target system. Also, the DDM files that refer to the remote files must all specify the same remote location information. If this restriction is not met, an error message is displayed to the user of PC Support/400 or to the user of the Open Query File (OPNQRYF) command who requested the query.

Sort Utility

The sort utility supports remote file processing with DDM anywhere that it supports local file processing for both AS/400 and non-AS/400 target systems.

Generally, on the Format Data (FMTDTA) command, DDM file names can be specified anywhere a database file name can be specified.

- A DDM file name can be specified on the SRCFILE parameter, and a member name can be specified on the SRCMBR parameter, for AS/400 or System/38 target systems. If the remote file referred to by the DDM file is not on an AS/400 system or a System/38, a member name cannot be specified.
- DDM file names can also be specified on the INFILE parameter (to access a remote file as the input file for conversion) or on the OUTFILE parameter (to access a remote file as the output file of the conversion). Both parameters cannot specify DDM file names at the same time.

Application Programs Considerations

The following AS/400 licensed programs support DDM for accessing remote files, with limitations:

- OfficeVision/400
- PC Support/400

Note: AS/400 Business Graphics Utility does not support DDM.

OfficeVision/400

OfficeVision/400 supports remote file processing using DDM for selected functions. The functions that support DDM files are:

- The Print Document (PRTDOC) command may use DDM files if the OUTFILE parameter is specified or if an output device of file is specified on the Print Options display. For more information, see "OUTFILE Parameter Considerations" on page 5-19.
- The get and get graphic functions of the OfficeVision/400 word processing function allow source and graphic data to be retrieved from DDM files. These functions are interactive and can seriously affect performance if large amounts of data are requested.

PC Support/400

The transfer function in PC Support/400 can be used with DDM to transfer data between a personal computer attached to a local AS/400 system and another remote system. When the transfer function is being used, the remote system must be an AS/400 system or a System/38. The PC Support/400 copy commands, Copy to PC Document (CPYTOPCD) and Copy from PC Document (CPYFRMPCD), can be used to copy data on a host system or between host systems.

Figure 2-4 shows a personal computer attached to the local AS/400 system. The PC Support/400 user can access data on remote systems through a DDM file defined on the local AS/400 system. The AS/400 system with the personal computer attached can only be the source system.

- The **PC Support/400 transfer function** can be used by a personal computer user to transfer data from a remote file to the personal computer, or to transfer data from the personal computer to a remote file. Only a personal computer user can start the requests, not an AS/400 user.

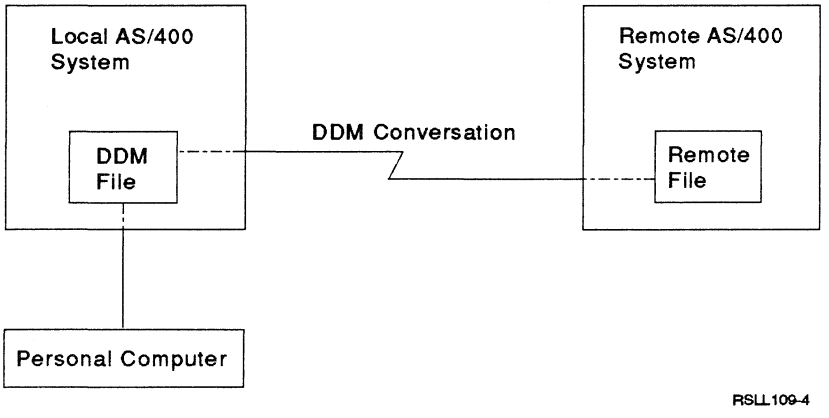


Figure 2-4. Using DDM with PC Support/400

- The **PC Support/400 copy commands** can be used with DDM to copy data from a personal computer document located on the local AS/400 system to a database file on the remote AS/400 system, or to copy data to a personal computer document on the local AS/400 system from a database file on the remote AS/400 system.

Note: For PC Support/400, database query allows accessing of multiple remote files (via DDM files) at the same time. For more information, see "Multiple Remote Files" on page 2-13.

PC Support/400 Transfer Function Considerations: A personal computer user can use the transfer function in PC Support/400 and the DDM support on the local AS/400 system to which the personal computer is attached either to transfer data *from* the personal computer to a remote file, or to transfer data from a remote file *to* the personal computer. The remote file must be on an AS/400 system or a System/38.

When DDM is used to transfer files or data from a remote system *to* an attached personal computer, the DDM files (that refer to remote files) on the local AS/400 system cannot be joined with local files to transfer data to the personal computer. (That is, data from files on both the remote and local systems cannot be joined.) However, a DDM file can specify a remote file that is a logical join file built over multiple physical files. DDM files that refer to the same target system and use the same remote location information can be joined.

A transfer request that requires group processing does not work if the local system is a System/38 and the remote system is an AS/400 system, or if the local system is an AS/400 system and the remote system is a System/38.

When DDM is used to transfer a file or data *from* an attached personal computer to a remote system, a remote file cannot be created on the target system. The remote file must already exist before the data from the personal computer can be transferred. However, because the target must be an AS/400 system or a System/38, a new member can be added in the remote file before personal computer data is transferred to that file member.

PC Support/400 Copy Command Considerations: The AS/400 CL command Copy from Personal Computer Document (CPYFRMPCD) used in PC Support/400 can be used to copy data *from* a document located either on an AS/400 system *to* a database file member located on the same AS/400 system or on a remote AS/400 system using DDM. The CL command Copy to Personal Computer Document (CPYTOPCD) can be used to copy data *from* a database file member on a local AS/400 system or a remote AS/400 system (using DDM) *to* a document on the local AS/400 system. The remote file can be on a target AS/400 system or a non-AS/400 system. To use these commands, specify the name of a DDM file on the:

- TOFILE parameter in the Copy from PC Document (CPYFRMPCD) command, to copy a personal computer document to an AS/400 physical file.
- FROMFILE parameter in the Copy to PC Document (CPYTOPCD) command, to copy a member from an AS/400 database file to a personal computer document in a folder.

The following restrictions apply to the CL copy commands for PC Support/400:

- For the CPYFRMPCD command, a remote file cannot be created on the target system (whether it is an AS/400 system or a non-AS/400 system). The remote file must already exist before the personal computer document data can be copied to it. However, if the target is an AS/400 system or a System/38, a new member can be created for the remote file before the personal computer document data is copied to that file member.
- The CPYFRMPCD and CPYTOPCD commands are AS/400 CL commands and cannot be entered at the DOS prompt from the personal computer.

Chapter 3. Preparing to Use DDM

This chapter describes the requirements for using DDM.

The following kinds of requirements must be met in various situations for OS/400 DDM to be used properly:

- Communications requirements
- Security requirements
- DDM file requirements

  Note: Before determining which files should be accessed using DDM, review "Performance Considerations" on page 6-10.
- High-level language (HLL) program modification requirements

Note: Programming requirements and considerations for control language (CL) commands and data description specifications (DDS) are covered in Chapter 5 and Chapter 6.

Communications Requirements

Each AS/400 system in the DDM network must have:

- The APPC/APPN support or the PC Support/400 licensed program installed and configured on the system. For complete information about configuring APPC/APPN, see the *OS/400* Communications Configuration Reference*, the *APPC Programmer's Guide*, and the *APPN Guide*. For information on configuring PC Support/400, see the *PC Support/400 Technical Reference for DOS and OS/2*.
- At least one Systems Network Architecture (SNA) communications line connection that uses synchronous data link communications (SDLC), token-ring network, Ethernet, or X.25 protocol.

The number of sessions that can be used for DDM conversations is not limited by DDM. The maximum is determined in the same manner as for any other APPC-related communications. For parallel sessions, the session maximum is specified in the mode. For single session devices, the session maximum is always one. The session values are described in the *APPC Programmer's Guide*.

Security Requirements

You can prevent intentional and unintentional access to the data resources of a system by the DDM user. Access to data in the DDM environment can be limited—or prevented altogether—by a system-level network attribute, the DDMACC parameter on the Change Network Attributes (CHGNETA) command on the system. This attribute allows the system (as a target system) to prevent all remote access; or it allows the system to control file access by using standard authority to files and, further, by using an optional user exit program to restrict the types of operations allowed on the files for particular users.

To provide adequate security, you may need to set up additional user profiles on the target system, one for each source system user who can have access to one or more target system files. Or, a default user profile should be provided for multiple source system users. The default user profile is determined by the communications entry used in the subsystem in which the target jobs are run.

See Chapter 4 for security information relating to DDM. For user profiles (or their equivalent) on non-AS/400 target systems, refer to that system's documentation.

DDM File Requirements

Before remote files can be accessed by an AS/400 system, DDM files must be created on the source system. See Chapter 5 for a description of the Create DDM File (CRTDDMF) command. At the time a DDM file is used, the device (remote location name) and mode (APPC session characteristics) specified in the DDM file must also exist on the system if APPN is not used. If APPN is used, then the device does not need to exist on the system. However, the system identified by the remote location name must exist within the APPN network. The APPN parameter on the Create Controller Description (APPC) (CRTCTLAPPC) and the Create Con-

troller Description (SNA Host) (CRTCTLHOST) commands controls whether or not APPN is used.

Program Modification Requirements

Remote files can be accessed by AS/400 application programs written in the HLL and control language. In most cases, these applications can access both local or remote files without the programs being changed. However, some considerations and restrictions may require the programs to be changed and recompiled. These are grouped in three categories:

- AS/400 functions that are not supported by the DDM architecture, but for which a System/38 extension to the architecture may exist. These functions can be used only when the source and target systems are System/38s or AS/400 systems.
- Restrictions and considerations that apply when the *source* or *target* system is an AS/400 system.
- Restrictions and considerations that apply to all target systems (AS/400 systems and non-AS/400 systems). User programs accessing local files should program for abnormal conditions such as *No record found*, *End of file*, and *Record lock time-out on read for update*. These conditions can also occur when a remote file is being accessed using DDM. In addition, the use of DDM exposes the program to communication line failures while sending disk I/O operations.

  When a communications failure occurs, the system sends an appropriate message to the job, which is returned to the application program as a generic file error. Each high-level language provides unique user syntax capabilities for user-controlled handling or default processing of exceptional results of a disk operation. Some languages may permit the user to retrieve the job message identification (ID) that would specifically indicate a DDM communications failure. Refer to the appropriate language manual for specific capabilities.

  For secondary SDLC lines, it is recommended that the INACTTMR parameter of the Create Line Description (SDLC) (CRTLINSDLC) command be set on the source and target systems to detect the stopping of polling by the primary system. This prevents the possibility of a DDM read-for-update record lock lasting indefinitely due to a communications failure on the primary system.

The restrictions and considerations relating to each of these groups are described in the following sections.

DDM Architecture-Related Restrictions

The following items are DDM architecture-related restrictions. Therefore, application programs that use these items may have to be changed and recompiled before they can access remote files:

- For more information about how commitment control is supported by the DDM architecture, see "Commitment Control Support" on page 2-4.
- The DDM architecture does not support AS/400 multiformat logical files. However, because multiformat logical files are supported as a System/38 extension to the DDM architecture, they can be used with DDM, but only if the source and target systems are AS/400 systems or System/38s.
- Externally described data (using data description specifications [DDS] on an AS/400 system) is not supported by the DDM architecture. However, DDS can still be used, especially if both systems are AS/400 systems or System/38s. If the target system is an AS/400 system or a System/38, most of the DDS support can be used as though the remote file is a local file. For the DDS considerations and limitations when DDM is used, see "Data Description Specifications (DDS) Considerations" on page 5-22.
- To access folder management services objects, the source system must support Level 2.0 or Level 3.0 of the DDM architecture for stream files and the stream access method. The following restrictions for the byte stream model apply:
  - WAIT time is not supported by the folder management services on the Lock Data Stream (LCKSTR) command. The user

must handle the waiting function on the source system.

  - The Copy File (CPYFIL) command used to copy a document on an AS/400 system is supported with the restrictions noted in Appendix D. Only the header information is copied; no data is copied.
  - The DELDRCOP (DRCALL) parameter is not supported on the Delete Directory (DELDRC) command.

- Personal computer generic names are not allowed when performing operations on data management objects such as files, libraries, or members. However, generic names are allowed when performing operations on folder management services objects such as documents and folders. Generic names are supported where the personal computer supports the operation and in the manner that the personal computer supports the operation. For example, generic names are not supported for folders using the rename and delete commands because the personal computer does not support them.

AS/400 Source and Target Restrictions and Considerations

When the *source* system is an AS/400 system, AS/400 database functions can be used on remote files, with the following restrictions:

- A source AS/400 system can create files on a System/38, but the DDM architecture file models are used. As a result, no multiformat logical or join logical files can be created on a non-AS/400 target system, including a System/38.
- Save or restore operations do not save or restore the data on a target system; only the DDM file object can be saved or restored locally.
- Operations that delay for a time period (that is, that wait for a file or record) are determined by the time values specified on the target system. (These values are specified by the WAITFILE and WAITRCD parameters on various CL commands.) This can result in increased delay times when DDM is used to access files or records remotely. For more information about these values, see the *Database Guide*.
- Query requests (OPNQRYF) to a System/38 cannot use group selection and join processing.
- When running System/36 applications to or from an AS/400 system, these applications may result in time-outs while waiting for a resource to become available. When running System/36 applications to or from another System/36, the application waits indefinitely for the resource to become available.

  For both source and target DDM jobs, due to the way DDM sends APPC operations, it is possible for the DDM job on the secondary side of the APPC conversation to wait indefinitely after a line failure or other failures at the remote system.

  Consider the following suggestions to avoid indefinite waits:

  - If the remote system supports record lock time-outs, ensure reasonable time values are specified. For example, on a target AS/400 system or System/38 database file, do not use maximum values for CRTPF ... WAITRCD.

    WAITRCD addresses read-for-update operations, but does not apply to other file operations, such as read only, add, and so on.
  - When using an SDLC secondary line, use a time value for the line inactivity timer (INACTTMR). Do *not* use the *NOMAX value. See the *Communications Management Guide* for additional information on an SDLC line description.
  - Provide the person responsible for system operation with the associated line, controller, and device names (or a list of DDM jobs that may run). If a DDM job then appears to be waiting indefinitely, this person could display the job information to determine if the job is waiting indefinitely by reviewing the job's processing unit time use (by using the Display Job (DSPJOB) command to display the active run attributes).

When the *target* system is an AS/400 system, AS/400 database functions can be used to access remote files, with the following restrictions:

- The physical files that the logical files or join logical files are based on must exist on the same AS/400 system. See the *Database Guide* for complete information about join and nonjoin logical files, and see the *DDS Reference* for descriptions of the DDS keywords PFILE and JFILE.
- A logical file on a source AS/400 system cannot share the access path of a remote file (on any target system).
- Query requests (OPNQRYF), which require group selection and join processing from a System/38, do not work.

Non-AS/400 Target Restrictions and Considerations

In addition to the restrictions that apply when the target system is an AS/400 system, the following restrictions also may apply when the target system is not an AS/400 system or a System/38. Whether they apply depends on what the target system supports. You should refer to that system's documentation for more information.

- Only field data types that are common to the source and target systems can normally be processed by HLL applications. Floating-point data is an example of a data type that may not be common. Records can be transmitted that contain floating-point data, but the representation of floating-point data sent between systems may differ.

  The packed signs sent between systems may differ; for example, one system may use a C and another system may use an F.

  Note: It is possible for you to write your application program so that it interprets the byte string for a record processed through a DDM file in any way that you wish. However, whenever you do this, it is your responsibility to ensure that the data is handled correctly.
- Any operations that request a delay period before returning, such as for record lock wait times, may be rejected or changed to a zero wait time by the target system.
- Lock requests may be changed by the target system to a more restrictive lock. This may prevent some operations from occurring at the same time that could otherwise be performed on the local AS/400 system. See "ALCOBJ (Allocate Object) Command" on page 5-9 for more information.
- Some AS/400 parameters are ignored or cause errors if they are used during remote file processing on non-AS/400 target systems. Examples are the FRCRATIO and FMTSLR parameters on some of the file commands. For more information, see "OVRDBF (Override with Database File) Command" on page 5-16 and see "CPYF (Copy File), CPYSRCF (Copy Source File), and CPYFRMQRYF (Copy from Query File) Commands" on page 5-11.
- Member names are not supported in the DDM architecture. When the target system is not an AS/400 system or a System/38, CL commands that have a MBR parameter, such as the Clear Physical File Member (CLRPFM) command, must be changed if the parameter specifies a member name that is different than the file name. If the member name is different, an error occurs if the command is used for a non-AS/400 remote file. For some commands, MBR(*FIRST) or MBR(*LAST) is also valid. See "Member-Related Commands" on page 5-21 for a list of all the CL commands related to file members, and for those that are not valid for accessing files on non-AS/400 target systems.

  Note: MBR(*LAST) is not supported by System/38.
- If a parameter on a CL command requires the name of a source file, then the names of the DDM files that refer to non-AS/400 target files cannot be specified. An AS/400 system cannot determine whether a remote file on a non-AS/400 target is in fact a source file. (See "Source File Commands" on page 5-22 for a list of all the CL commands related to source files.)
- Certain AS/400 commands that are valid for AS/400 or System/38 target systems are not valid for other targets. See "DDM-Related CL Command Lists" on page 5-19 for the lists of commands that are not supported when the target is not an AS/400 system or a System/38.

Chapter 4. Security Considerations for DDM

This chapter describes how AS/400 security relates to DDM and how it can limit access to the data resources of a target system by source system programs and users. The access to target AS/400 data can be limited by using standard authority to files, standard authority to commands, and an optional user exit program in the DDM environment at the target system.

Security checking is performed when a remote user accesses an AS/400 file. The remote user must be authorized to perform the operation (open, close, read, or write, for example) or the DDM request is rejected. Application programs on the AS/400 system can be isolated from each other by object authorities. For more information about source and target system security when APPC is being used (as with DDM), see the *APPC Programmer's Guide*. For details about target system security when you are using PC Support/400, see the *PC Support/400 Technical Reference for DOS and OS/2*.

Elements of DDM Security

When DDM is used, the data resources of each system in the DDM environment should be protected. This is done using three groups of security elements that are controlled by the following parameters:

- For system-related security, **LOCPWD parameter** on the configuration commands is used on each AS/400 system to indicate the system validation password to be exchanged between the source and target systems when an APPC communications session is first established between them. Both systems must exchange the same password before the session is started. (On System/36, this password is called the location password. The password that the target System/38 uses is in its device description for the source system.) The password that the AS/400 system uses is in its remote location configuration.
- For user-related security, **SECURELOC parameter** on the configuration commands is used on each AS/400 system to indicate whether it (as a target system) accepts incoming access requests that have their security already verified by the source system. The SECURELOC parameter value is specified in the remote location configuration. The SECURELOC value can be specified differently for each remote location.

  The SECURELOC parameter is used with the following security elements (for which more information is given in the topics "DDM Source System Security" on page 4-2 and "DDM Target System Security" on page 4-2):

  - The user ID sent by the source system, if allowed by this parameter
  - The target system user profiles, including default user profiles
- For object-related security, the **DDMACC parameter** is used on the Change Network Attributes (CHGNETA) command to indicate whether the files on the AS/400 system can be accessed at all by another system and, if so, at which level of security the incoming requests are to be checked. More information about this object-related parameter is provided in the topic "DDM Network Attribute (DDMACC Parameter)" on page 4-3.

  - If *REJECT is specified on the DDMACC parameter, all DDM requests received by the target AS/400 system are rejected.
  - If *OBJAUT is specified on the DDMACC parameter, normal object-level security is used on the target system.
  - If the name of an optional, user-supplied user exit program (or access control program) is specified on the DDMACC parameter, an additional level of security is used. The user exit program can be used to control whether a given user of a specific source system can use a specific command to access (in some manner) a specific file on the target system. (See the topic "User Exit Program for Additional Security" on page 4-4 for details.)
  - When a file is created on the target system using DDM, the library name specified contains the file. If no library name is specified on the DDM request, the current library (*CURLIB) is used. The file authority defaults to allow only

the user who created the file or the target system's security officer to access the file.

Most of the security controls for limiting remote file access are handled by the target system. Except for the user ID provided by the source system, all of these elements are specified and used on the target system. The source system, however, also limits access to target system files by controlling access to the DDM file on the source system and by sending the user ID, when needed, to the target system.

DDM Source System Security

The first area of source system security is with the DDM file itself. When the DDM file is created by the Create DDM File (CRTDDMF) command, the AUT parameter is used to control what rights of use all users on the *source* system have for the DDM file. The AUT parameter can allow all (or none) of the source system users to use the DDM file to access a remote file, and it can specify how all users are authorized to use the DDM file itself.

Once the DDM file is created, the Grant Object Authority (GRTOBJAUT) command or the Revoke Object Authority (RVKOBJAUT) command can be used to explicitly grant (or revoke) rights to specific users for the DDM file's use. The AUT parameter and these commands work the same for DDM files as for any other created OS/400 object.

The AS/400 system, as a source system, never sends a user password when starting the TDDM on the target system. (System/36 sends no user password either.) If the source system security is considered sufficient, the target system can specify that user IDs should be sent; if not, no user ID is sent. On the AS/400 system, this is dictated by the SECURELOC parameter value in effect on the *target* system; this parameter is specified in the target system's remote location configuration.

- If SECURELOC(*YES) is specified, it indicates that the target system accepts the source system security procedures; the source system, on each program start request operation, sends the user ID and the already verified indicator. The user ID is compared to those in the user profiles on the target system to verify the source system user's right for access.
- If SECURELOC(*NO) is specified, it indicates that the target system does not accept the source system security procedures. No user ID is sent; a default user profile on the target system must be created and used to verify the right for access.

  Additional security can be provided on the target system if a user exit program is written and used to restrict each source system user that attempts to access its files or to perform other functions via commands submitted on the Submit Remote Command (SBMRMTCMD) command.

Note: DDM does not allow the target system (as a target) to make requests, so the source system is implicitly secure from the target.

DDM Target System Security

When the target system is an AS/400 system, several elements used together determine whether a request to access a remote file is allowed or not:

User-related security elements: The SECURELOC parameter on the target system, the user ID sent by the source system (if allowed), and a user profile or default user profile on the target system.

Object-related security elements: The DDMACC parameter and, optionally, a user exit program supplied by the user to supplement normal object authority controls.

User-Related Elements of Target Security

The value specified for the SECURELOC parameter in the target system's remote location configuration of a source system determines whether a user or program on the source system is to supply a user ID that has already been verified by the source system. On the target system for the remote location configuration:

- If SECURELOC(*YES) is specified, the source system sends the user ID of the user requesting remote system access, and the target system verifies that it exists in a user

profile. If the user ID matches a user profile on the target system, a job is started to handle the remote file access requests from the source system user. If no user profile exists for the user ID that was sent, or if the user ID is not valid, the initial access request is rejected, and an error message is sent both to the source system user and to the system operator message queue on the target system. (The message sent to the source system user is different than the target system message.)

- If SECURELOC(*NO) is specified, no user ID is sent by the source system and the target system must have a default user profile to initiate the target system job. The contents of this profile are controlled by target system personnel. Examples of items that it should contain are: the names of libraries, objects, and commands on the target system that can be used.

  The name of the default user profile must be specified on the DFTUSR parameter of the Add Communications Entry (ADDCMNE) command on the target system; this command adds a communications entry to the subsystem description used for the target system job. If SECURELOC(*NO) is specified and no default profile exists, the initial access request is rejected.

When the target system is an AS/400 system, the user profiles associated with the target jobs must be authorized to use CL commands before equivalent DDM requests can be performed. See Chapter 5 and Appendix D for more information on the CL commands for which user profiles must be authorized. The local user's authorization to commands does not affect authorization on the target system.

Target Jobs and User Profiles: The AS/400 system creates a separate target job for each different remote system user (that is, for each separate program start request operation received from source systems). Separate jobs are also created for different users from the *same* system. Before any operations can be performed on target system database files in a job, the user profile associated with the target job must be specifically authorized to use each of the files for which access has been requested by a user in the source job. In addition, the user profile needs to be authorized to the AS/400 commands equivalent to the DDM function being requested by a user in the source job.

When a user ID is sent by a source system, the associated user profile is used for the target job. If there is no user profile for that user ID, the DDM request is not allowed. A user profile should be created on the target system for every source system user of DDM.

Object-Related Levels of Target Security

When the AS/400 system is a target system, there are three different object-related levels at which security can be enforced to control access to its database files: The system can be secured to prevent all DDM requests from accessing its files, it can use normal object authorization support to determine which users can access what files, or it can combine normal object authorization support with a user exit program written by the user to further restrict file access. The system-level DDMACC parameter determines which of the three levels is used.

DDM Network Attribute (DDMACC Parameter): The network attribute parameter DDMACC (DDM access) is used to determine how the AS/400 system, as a *target* system, processes requests from other systems. This parameter is initially set to *OBJAUT. As explained later, the value of this parameter can be changed by the Change Network Attributes (CHGNETA) command.

The values for the DDMACC parameter are:

***SAME** Specifies that the current value of the DDMACC parameter remains unchanged. This is the default value on the CHGNETA command for each AS/400 system.

***REJECT** Specifies the system will not allow any DDM requests from remote systems. However, this system (as a source system) can still use DDM to access files on other systems that allow it. No system can access files on any AS/400 system that specifies *REJECT.

If *REJECT is specified while DDM is already in use, all *new* jobs on any source system requesting

access to this system's files are rejected and an error message is returned to those jobs; existing jobs are not affected.

***OBJAUT** All remote requests are allowed, but they are controlled by the object authorizations on this system (normal AS/400 object level security). For each file on the system, all users, no users, or only specific users (by user ID) can be authorized to access the file. If SECURELOC(*YES) is specified, specific (or multiple) user profiles can be authorized to the file. Otherwise, the authorizations must be given in the default user profile identified in the communications entry (on the ADDCMNE command). See the *Database Guide* for more information about object authority. This is the value shipped with the system.

When the value *OBJAUT is specified, it indicates that no further verification (beyond AS/400 object level security) is needed.

qualified-program-name

Specifies the name of the user exit program supplied by the user (and the library in which it is stored) that can *supplement* AS/400 object level security (which still applies). This user exit program is passed a parameter list, built by the target system, that identifies the source system user and the request. The program is used to determine whether to allow the request. See the topic "User Exit Program for Additional Security" for more information.

Any error occurring while using or attempting to use this user exit program sends an error message to the source system. If the source system is an AS/400 system or a System/38, the message might indicate (for example) that the user exit program was not found, the user was not authorized to use it, or that the number of parameters was sent in the parameter list for the user exit program is not valid.

For a description of the DDMACC parameter, see the Change Network Attributes (CHGNETA) command described in the *Communications Management Guide*.

Changing the DDMACC Network Attribute: The DDMACC parameter, initially set to *OBJAUT, can be changed to one of the previously described values by using the Change Network Attributes (CHGNETA) command, and its current value can be displayed by the Display Network Attributes (DSPNETA) command. You can also get the value in a CL program by the Retrieve Network Attributes (RTVNETA) command.

If the DDMACC parameter value is changed, although it takes effect immediately, it affects only *new* DDM jobs started on this system (as the target system). Jobs running on this target system before the change was made continue to use the old value.

User Exit Program for Additional Security

Customers who use menu-level security (described in the *Database Guide*), done by restricting the end user's access to functions on the system, are likely to have a large number of public files (those files that are created to which the public has some or all authority). A user exit program can be written and used to restrict each DDM user's access to public files and to private files; the name of the program must be specified on the DDMACC parameter of the Change Network Attributes (CHGNETA) command.

The user exit program must exist on the target system. The target DDM support calls this program:

- For each user's *initial* reference to a file to verify whether the user can have access to the file. When a file is referred to for I/O operations, this verification occurs only once, when the file is opened. The user exit program indicates to the TDDM whether the access request is accepted or rejected.
- For each of the other functions listed in the *Subapplication* field of the table in Figure 4-1 on page 4-6.

When a user exit program is specified, the TDDM first checks for errors in the access request received from the source system. If no errors are detected, the TDDM builds the parameter list, calls the user exit program, and passes the parameter list to it.

User Exit Program Requirement: The purpose of the user exit program created by the user is to determine whether a user's access request is to be accepted or rejected. It does so using the values passed to it in the parameter list. The program can be written to verify all the values in the parameter list, or to verify part of them. The program *must* return a return code of 1 to indicate that the request is accepted, and it *should* return a 0 to indicate that the request is rejected.

User Exit Program Parameter List: The user exit program on the target system passes two parameter values (a character return code field and a character data structure containing various parameter values, which are shown in Figure 4-1 on page 4-6). The user exit program on the target system uses the character data structure parameter values, that are passed by the TDDM, to evaluate whether to allow the request from the source system. The parameter list is created each time a file access request or command request is sent to the TDDM; when any one of the functions shown for the *Subapplication* field is requested, the parameter list is created. When file I/O operations are performed, this parameter list is created only for the file open request, not for any of the I/O operation requests that follow.

The program uses the parameter list to determine whether a source system user's file access or command request should be accepted or rejected. The list contains the following parameters and values:

- The name of the user profile or default user profile under which the source system user's request is run.
- The name of the application program on the source system being used. For DDM use, the name is *DDM.
- The name of the command or function (subapplication) being requested for use on the target system or one of its files.

  Most of the functions listed in Figure 4-1 on page 4-6 directly affect a file, including the EXTRACT function, which extracts information from the file when commands such as Display File Description (DSPFD) or Display File Field Description (DSPFFD) are specified by the source system user. Some functions are member-related functions, such as the CHGMBR function, which allows characteristics of a member to be changed. The COMMAND function indicates that a command string is submitted by the Submit Remote Command (SBMRMTCMD) command to run on the target system.
- The name of the file (object) to be accessed in the way specified on the previous parameter. This field does not apply if a command string (COMMAND) or stream and directory access commands are being submitted.
- If the stream and directory access commands are specified, then the object and directory fields have a value of *SPC. The user must go to the *Other* field to get the alternative object name and alternative path name.
- The name of the library containing the file, if a file is being accessed.
- The name of the file member, if a file member is being accessed. Stream and access commands have a value of *N.
- The format field does not apply for DDM.
- The length of the next field, which varies, depending on how it is used.
- The *Other* field is used for as many as three of the following six values; the first two are always specified (*N may be used for the second value if the system name cannot be determined), and either of the last four may be specified, depending on the type of function specified in the *Subapplication* field.
  - The location name of the source system. This matches the RMTLOCNAME parameter value specified in the target system's device description for the source system.
  - The system name of the source system.
  - If a file was specified and it is to be opened (OPEN) for I/O operations, this field indicates which type of operation is being requested. For example, if a file is

being opened for read operations only, the input request value is set to a 1 and the remaining values are set to a 0.

- The alternative object name.
- The alternative directory name.
- The name of the AS/400 command, if a command string is being submitted, followed by all of its submitted parameters and values.

Examples of a user exit program and a parameter list follow Figure 4-1.

Figure 4-1 (Page 1 of 2). Parameter List for User Exit Program on Target System

Field	Type	Length	Description
User	Character	10	User profile name of target DDM job.
Application	Character	10	Application name: '*DDM ' for Distributed Data Management.
Subapplication	Character	10	Requested function: 'ADDMBR ' 'DELETE ' 'RENAME ' 'CHANGE ' 'EXTRACT ' 'RGZMBR ' 'CHGMBR ' 'INITIALIZE' 'RMVMBR ' 'CLEAR ' 'LOAD ' 'RNMMBR ' 'COMMAND ' 'LOCK ' 'UNLOAD ' 'COPY ' 'MOVE ' 'CREATE ' 'OPEN '
Object	Character	10	Specified file name. *N is used when the subapplication field is 'COMMAND '. *SPC is used when the file is a document or folder.
Directory	Character	10	Specified library name. *N is used when the subapplication field is 'COMMAND '. *SPC is used when the library is a folder.
Member	Character	10	Specified member name. *N is used when the member name is not applicable.
Format	Character	10	Not applicable for DDM.
Length	Decimal	5,0	Length of the next field.

Figure 4-1 (Page 2 of 2). Parameter List for User Exit Program on Target System

Field	Type	Length	Description
Other	Character	—	The length of the structure varies, depending on which of the following items is used.
		10	Remote location unit name of source system.
		10	System name of the source system. If this value is not available, this field contains '*N
			The following varies, depending on the function:
			If OPEN is specified to open a file:
		1	Input request Char(1) 1=yes 0=no
		1	Output request Char(1) 1=yes 0=no
		1	Update request Char(1) 1=yes 0=no
		1	Delete request Char(1) 1=yes 0=no
		12	Alternative object name.
		63	Alternative directory name.
		2000	The command string if COMMAND is specified to submit a command (maximum length of 2000 characters).

*N = null value indicates a parameter position for which no value is being specified, allowing other parameters to follow it in positional form.

User Exit Program Example: The following user exit program represents the source code for a PL/I program created by a security officer on a remote system in Chicago. To define this user exit program to the system, the security officer specifies the following:

```
CHGNETA DDMACC(DJWLIB/$UEPGM)
```

where DJWLIB/$UEPGM is the qualified name of the user exit program.

Because the security officer wants to specifically prevent user KAREN from opening file RMTFILEX, the user exit program returns a 0 in the return code field when she attempts to open file RMTFILEX; the user exit program returns a 1 in the return code field in all other cases indicating that requests by other users are permitted.

```
$UEPGM: PROCEDURE (RTNCODE,CHARFLD);
DECLARE
      RTNCODE  CHAR(1);
DECLARE
 1  CHARFLD,
    2  USER     CHAR(10),
    2  APP      CHAR(10),
    2  FUNC     CHAR(10),
    2  OBJECT   CHAR(10),
    2  DIRECT   CHAR(10),
    2  MEMBER   CHAR(10),
    2  RESERVED CHAR(10),
    2  LNGTH    PIC '99999',
    2  LUNAME   CHAR(10),
    2  SRVNAME  CHAR(10),
    2  OTHER,
       3  INRQS    CHAR(1),
       3  OUTRQS   CHAR(1),
       3  UPDRQS   CHAR(1),
       3  DELRQS   CHAR(1),
       3  ALTOBJ   CHAR(12),
       3  ALTDIR   CHAR(63),
       3  REMAING  CHAR(1921);
DECLARE
  OPEN   CHAR(10) STATIC INIT('OPEN'),
  KAREN  CHAR(10) STATIC INIT('KAREN'),
  RMTFILEX CHAR(10) STATIC INIT('RMTFILEX');

DECLARE
  ZERO  CHAR(1)  STATIC INIT('0'),
  ONE   CHAR(1)  STATIC INIT('1');

IF (FUNC   = OPEN    ) &
   (USER   = KAREN   ) &
   (OBJECT = RMTFILEX)
THEN
  RTNCODE = ZERO;
ELSE
  RTNCODE = ONE;
END $UEPGM;
```

Parameter List Example: The following commands are in a CL program that a user named KAREN on the source system (NEWYORK) is using. The remote location configuration of the target system (CHICAGO) specifies SECURELOC(*YES) for the NEWYORK source system, indicating that user IDs are to be sent and that a user profile for KAREN exists on the target system.

The program used by KAREN accesses a DDM file named LOCFILEX that opens a remote file named RMTFILEX on the target system in Chicago. Both systems are AS/400 systems. The file is being opened for input.

```
CRTDDMF  FILE(LOCFILEX)  RMTFILE(LIBX/RMTFILEX)
         RMTLOCNAME(CHICAGO)
  :
OPNDBF   FILE(LOCFILEX)  OPTION(*INP)
MONMSG   MSGID(CPF0000)    EXEC(GOTO EXIT)
  :
CLOF   OPNID(LOCFILEX)
EXIT:  ENDPGM
```

When the Open Database File (OPNDBF) command is run on the NEWYORK source system, the DDM file named LOCFILEX is opened, and DDM sends a request to the target system to open RMTFILEX in LIBX for input operations. From this information, the target system builds the following parameter list to be used by the user exit program for verification:

```
KAREN *DDM OPEN RMTFILEX LIBX *N 0 24 CHICAGO NEWYORK 1000
```

This parameter list shows only the significant characters that would be sent in each field; all the padded blanks and zeros are not shown. For example, the field containing KAREN would be padded with five blanks because it is a 10-character field. This parameter list is sent only for the open operation, although several input operations may be performed on RMTFILEX.

This parameter list is sent to the user exit program specified on the DDMACC parameter of the Change Network Attributes (CHGNETA) command. The user exit program determines if user KAREN is authorized to open RMTFILEX. If she is authorized, the program returns a 1 in the return code field, and she can open the file and perform read operations. If the program returns a 0 in the return code field, user KAREN receives a message in the job log indicating that she is not authorized to use the file.

When all the input operations are completed, the Close File (CLOF) command runs on the source system, and DDM sends the request to close the file.

User Exit Program Considerations: If the user exit program is a CL program that creates an OS/400 exception, an inquiry message is sent to the system operator on the target system if, for the target job, the job attribute INQMSGRPY is *RQD (the default) or *SYSRPYL with no value in the reply list for this message. The user exit program waits for a response to the message on the target system, which causes the source job to wait also.

There are other potential situations in which waiting could occur. For example, if lengthy wait values are specified on the WAIT parameter of the Allocate Object (ALCOBJ) or Receive Message (RCVMSG) command, both the source and target jobs wait up to the maximum time specified for an object lock to be obtained or a message to be received by the target job.

Chapter 5. CL Command Descriptions and DDS Considerations for DDM

This chapter contains DDM-related information about specific AS/400 control language (CL) commands, data description specifications (DDS) considerations, DDS keywords, and DDM user profile authority.

Refer to the *CL Reference* manual for further information about the command descriptions and syntax diagrams.

Described are:

- DDM-specific CL commands
- DDM-related CL commands, containing only information relating to DDM
- DDM-related parameter considerations, providing information about specific CL command parameters affected by DDM
- Command lists, showing various groupings of all the DDM-related CL commands
- DDS specifications, providing only DDM-related DDS considerations and DDS keywords
- DDM user profile authority, for use with a remote system

DDM-Specific CL Commands

The DDM-specific CL commands include:

- Change DDM File (CHGDDMF)
- Create DDM File (CRTDDMF)
- Display DDM Files (DSPDDMF)
- Reclaim DDM Conversations (RCLDDMCNV)
- Submit Remote Command (SBMRMTCMD)
- Work with DDM Files (WRKDDMF)

CHGDDMF (Change DDM File) Command

The Change DDM File (CHGDDMF) command changes one or more of the attributes of a DDM file on the local (source) system. The DDM file is used as a reference file by programs on the AS/400 source system to access files located on any target system in the OS/400's DDM network.

To use this command, you can enter the command as shown in the following example or select option 2 (Change DDM File) from the Work with DDM Files display. For further information about using the menu options, see the topic "WRKDDMF (Work with DDM Files) Command" on page 5-6.

Example

```
CHGDDMF  FILE(SOURCE/SALES)  MODE(MODEX)
```

This command changes the communications mode for the DDM file named SALES stored in the SOURCE library on the source system; the mode is changed to MODEX.

CRTDDMF (Create DDM File) Command

The Create DDM File (CRTDDMF) command creates a DDM file on the local (source) system. The DDM file is used as a reference file by programs on an AS/400 system to access files located on any remote (target) system in the AS/400's DDM network. Programs on the local AS/400 system know a remote file only by the DDM file's name, not the remote file's actual name. (The DDM file name, however, can be the same as the remote file name.)

The DDM file is also used when a CL command is submitted to the remote system. (The Submit Remote Command (SBMRMTCMD) command is used to submit the CL command, and the remote system must be an AS/400 system or a System/38.) When the SBMRMTCMD command is being used, the remote file normally associated with the DDM file is ignored.

The DDM file contains the name of the remote file being accessed and the remote location information that identifies a remote (target) system where the remote file is located. It can also specify other attributes that are used to access records in the remote file.

To use this command, you can enter the command as shown in the following examples or select F6 (Create DDM file) from the Work with DDM Files display. For further information about using the menu options, see the topic "WRKDDMF (Work with DDM Files) Command" on page 5-6.

Examples

- **Creating a DDM file to access a file on a System/38:**

```
CRTDDMF  FILE(SOURCE/SALES)  RMTFILE(*NONSTD 'SALES.REMOTE')
   RMTLOCNAME(NEWYORK)
```

  This command creates a DDM file named SALES and stores it in the SOURCE library on the source system. This DDM file uses the remote location NEWYORK to access a remote file named SALES stored in the REMOTE library on a System/38 in New York.

- **Creating a DDM file to access a file *member* on an AS/400 system:**

```
CRTDDMF  FILE(SOURCE/SALES)  RMTLOCNAME(NEWYORK)
   RMTFILE(*NONSTD 'REMOTE/SALES(APRIL)')
```

  This command creates a DDM file similar to the one in the previous example, except that now it accesses the member named APRIL in the remote SALES file stored in the REMOTE library on an AS/400 system.

- **Creating a DDM file to access a file on a System/36:**

```
CRTDDMF  FILE(OTHER/SALES)  RMTFILE(*NONSTD 'PAYROLL')
   RMTLOCNAME(DENVER)  LVLCHK(*NO)
```

  This command creates a DDM file named SALES, and stores it in the library OTHER on the source system. The remote location DENVER is used by the DDM file to access a remote file named PAYROLL on a System/36 in Denver. No level checking is performed between the PAYROLL file and the application programs that access it. Because the ACCMTH parameter was not specified, the access method for the target system is selected by the source AS/400 system when the DDM file is opened to access the remote file.

Additional Considerations: For additional information about using advanced program-to-program communications (APPC) with DDM, refer to *APPC Programmer's Guide*.

DSPDDMF (Display DDM Files) Command

The Display DDM Files (DSPDDMF) command displays the details of a DDM file.

To use this command, you can type the command or select option 5 (Display details) from the Work with DDM Files display. For further information about using the menu options, see the topic "WRKDDMF (Work with DDM Files) Command" on page 5-6.

RCLDDMCNV (Reclaim DDM Conversations) Command

The Reclaim DDM Conversations (RCLDDMCNV) command is used to reclaim all DDM source system conversations that are not currently being used by a source job, even if the DDMCNV attribute value for the job is *KEEP. The command allows the user to reclaim unused DDM conversations without closing all open files or doing any of the other functions performed by the Reclaim Resources (RCLRSC) command.

The RCLDDMCNV command applies only to the DDM conversations for the job on the *source* system in which the command is entered. For each DDM conversation used by the source job, there is an associated job on the target system; the target job ends automatically when the associated DDM conversation ends.

Although this command applies to *all* DDM conversations used by a job, using it does *not* mean that all of them will be reclaimed. A conversation is reclaimed *only* if it is not being actively used. For the conditions under which the conversation is considered unused, see "Controlling DDM Conversations" on page 6-8.

SBMRMTCMD (Submit Remote Command) Command

The Submit Remote Command (SBMRMTCMD) command submits a CL command via DDM to run on the target system. The remote location information in the DDM file is used to determine the communications line to be used, and thus, indirectly identifies the target system that is to receive the submitted command.

Note: The remote file normally associated with the DDM file is not involved when the DDM file is used for submitting commands to run on the target system.

The SBMRMTCMD command can be used to send CL commands (and only CL) to an AS/400 system or a System/38. (It cannot be used to send non-AS/400 commands; for example, operation control language [OCL] commands cannot be sent to a target System/36.) The command

must be in the syntax of the target system (an AS/400 system or a System/38). The SBMRMTCMD command can submit any CL command that can run in both the batch environment and via the QCAEXEC system program. That is, if a command has values of *BPGM *and* *EXEC specified for the ALLOW attribute, which you can display by using the Display Command (DSPCMD) command, that command can be submitted by the SBMRMTCMD command. (The SBMRMTCMD command uses the QCAEXEC system program to run the submitted commands on the target system.) However, because some of these allowable commands require intervention on the target system and/or may not produce the results expected, you should consider the items listed in the topic "Restrictions" first.

- The primary purpose of this command is to allow a source system user or program to perform file management operations and file authorization activities on files located on a target AS/400 system or a target System/38. (For more information on file management operations, see "Performing File Management Functions on Remote Systems" on page 6-7.) The user must have the proper authority both for the CL command being submitted and for the target system objects that the command is to operate on. If the source system user has the correct authority to do so (as determined in a target system user profile), the following actions are examples of what can be performed on remote files using the SBMRMTCMD command:
  - Create or delete device files
  - Grant or revoke object authority to remote files
  - Verify files or other objects
  - Save or restore files or other objects

  Although the command can be used to do many things with files or objects, some are not as useful as others. For example, you could use this command to display the file descriptions or field attributes of remote files, or to dump files or other objects, but the output remains at the target system. (To display remote file descriptions and field attributes at the source system, use the Display File Description (DSPFD) and Display File Field Description (DSPFFD) commands with SYSTEM(*RMT) specified, and specify the names of the DDM files associated with the remote files.)

  For the commands that are considered useful when submitted by the SBMRMTCMD command to a target system, see Appendix B.
- A secondary purpose of this command is to allow a user to perform nonfile operations (such as creating a message queue) or to submit user-written commands to run on the target system.
- The CMD parameter allows you to specify a character string of up to 2000 characters that represents a command to be run on the target system.

Restrictions

1. Although remote file processing is synchronous within the user's job, which includes two separate jobs (one running on each system), file processing on the target system operates independently of the source system. Commands such as Override with Database File (OVRDBF), Override with Message File (OVRMSGF), and Delete Override (DLTOVR) that are dependent on the specific position of a program in a program stack (**recursion level**) or request level may *not* function as expected.

   For example, when multiple recursion levels that involve overrides at each level occur on the source system, and one or more overrides at a given level are submitted to the target system on the SBMRMTCMD command, the target system job has no way of knowing the level of the source system job. That is, a target system override can still be in effect after the source system override for a particular recursion level has ended.
2. Output (such as spooled files) created by a submitted command exists only on the target system. The output is *not* sent back to the source system.
3. Some types of CL commands should *not* be submitted to a target AS/400 system. The following are examples of types that are *not* the intended purpose of the SBMRMTCMD command and that may produce undesirable results:
   - All of the OVRxxxF commands that refer to database files, message files, and

device files (including communications and save files).

- All of the DSPxxxx commands, because the output results remain at the target system.
- Job-related commands like Reroute Job (RRTJOB) that are used to control a target system's job. The Change Job (CHGJOB) command, however, *can* be used.
- Commands that are used to service programs, like Service Job (SRVJOB), Trace Job (TRCJOB), Trace Internal (TRCINT), or Dump Job (DMPJOB).
- Commands that may cause inquiry messages to be sent to the system operator, like Start Printer Writer (STRPRTWTR) or Copy to Diskette (CPYTODKT). (Pass-through can be used instead.)

4. Translation is not performed for any *immediate* messages created by the target system, because they are not stored on the system; the text for an immediate message is sent directly to the source system to be displayed. (For all other message types, the target system sends back a message identifier; the message text that exists on the source system for that message identifier is the text that is displayed. This message text is whatever the source system text has been translated to.)
5. A maximum of 10 messages, created during the running of a submitted command, can be sent by the target system to the source system. If more than 10 messages are created, an additional *informational* message is sent that indicates where the messages exist (such as in a job log) on the target system. If one of those messages is an *escape* message, the first nine messages of other types are sent, followed by the informational message and the escape message.
6. The only types of messages that are sent by the target system are completion, informational, diagnostic, and escape messages.

Examples: Submitting a command to create another DDM file on the remote system:

```
SBMRMTCMD  CMD('CRTDDMF  FILE(SALES/MONTHLY)
   RMTFILE(*NONSTD ''SALES/CAR(JULY)'')
   RMTLOCNAME(DALLAS)')  DDMFILE(CHICAGO)
```

This submitted command creates, on the target system identified by the information in the DDM file named CHICAGO, another DDM file named MONTHLY; the new DDM file is stored in a library named SALES on the system defined by DDMFILE CHICAGO. The new DDM file on the CHICAGO system is used to access a file and *member* on a different system named DALLAS. The accessed file is named CAR in the library SALES and the member name in the file is JULY.

Notice that this CRTDDMF command string contains *three* sets of single apostrophes: one set to enclose the entire command being submitted (required by the CMD parameter on the SBMRMTCMD command), and a double set to enclose the file and member named in the RMTFILE parameter. Because the use of *NONSTD requires that nonstandard file names be enclosed in a set of apostrophes, this second set of apostrophes must be doubled because it is within the first set of apostrophes.

Submitting a command to change *text* in a display file:

```
SBMRMTCMD  CMD('CHGDSPF  FILE(LIBX/STANLEY)
   TEXT(''Don''''t forget to pair apostrophes.'')')
   DDMFILE(SMITH)
```

This command changes the text in the description of the display device file named STANLEY stored in library LIBX. Because the submitted command requires an outside set of single apostrophes (for the CMD parameter), each single or double apostrophe normally required in the TEXT parameter for *local* system processing must be doubled again for *remote* system processing. The coding above produces a single apostrophe in the text when it is displayed or printed on the remote system.

Submitting a command to replace a library list on the remote system:

```
SBMRMTCMD  CMD('CHGLIBL  LIBL(QGPL QTEMP SALES EVANS)')
   DDMFILE(EVANS)
```

This command changes the user's portion of the library list being used by the target job associated with the DDM file named EVANS, which is being used by the source job in which this SBMRMTCMD command is being submitted. In that source job, if there are other open DDM files that specify the remote location information, this library list is used for them also.

Additional Considerations: Override use example: The DDMFILE parameter on the SBMRMTCMD command is used to determine which target system the command (CMD parameter) should be sent to. Overrides that apply to the DDM file (not the remote file) are taken into account for this function. For example, if a file override was in effect for a DDM file because of the following commands, which override FILEA with FILEX, then the target system that the Delete File (DLTF) command is sent to is the one associated with the remote location information specified in DDM FILEX (the values point to the DENVER system, in this case).

```
CRTDDMF  FILE(SRCLIB/FILEA)  RMTFILE(SALES/CAR)
   RMTLOCNAME(CHICAGO)
CRTDDMF  FILE(SRCLIB/FILEX)  RMTFILE(SALES/CAR)
   RMTLOCNAME(DENVER)

OVRDBF   FILE(FILEA)  TOFILE(SRCLIB/FILEX)
SBMRMTCMD  CMD('DLTF RMTLIB/FRED')  DDMFILE(SRCLIB/FILEA)
```

This SBMRMTCMD command deletes the file named FRED from the DENVER system.

DDM conversations: When a SBMRMTCMD command is run on the target system, it has a target system job associated with it. Successive SBMRMTCMD commands submitted using the same DDM file and DDM conversation may run in the same or different target system jobs, depending on the value of the DDMCNV job attribute. The value of the DDMCNV job attribute determines whether the DDM conversation is dropped or remains active when the submitted function has completed. If the conversation is dropped, the next SBMRMTCMD command runs using a different target job. If several commands are submitted, either DDMCNV(*KEEP) should be in effect, or display station pass-through should be used instead of DDM.

See the topic "DDM-Related Jobs and DDM Conversations" on page 1-15 for an explanation of how the system handles DDM conversations, and see "DDMCNV Parameter Considerations" on page 5-18 for a description of the DDMCNV job attribute.

Command syntax verifying: The syntax of the command character string being submitted by the CMD parameter is not verified by the source system. In the case of a user-defined command, for example, the command definition object may or may not exist on the source system.

Command running results: Because the submitted command runs as part of the target system's job, the attributes of that job (such as the library search list, user profile, wait times, and running priority) may cause a different result than if the command were run locally.

Error message handling:

- For errors detected by the target system when processing the submitted command, the source system attempts to send to the source system user the same error information that was created on the target system. However, if the source system does not have an equivalent message for the one created on the target system, the message sent to the source system user has the message identifier and is of the message type and severity that was created on the target system; the message text sent for the error is default message text.

  The target system messages can be viewed on the source system by using pass-through and either the Work with Job (WRKJOB) or Work with Job Log (WRKJOBLOG) command. If the target job ends, the messages are in the target system's output queue, where they can be displayed by the Display Output Queue (DSPOUTQ) command.

- If the SBMRMTCMD command is used to call a CL program on the target system, any escape message that is not monitored and is created by the program is changed into an inquiry message and is sent to the system operator. If you don't want the target system operator to have to respond to this inquiry message before the job can continue, you can refer to the *CL Reference* manual and do either of the following on the target system:

  - If you want to specify a default reply for a specific *job*, you can use the INQMSGRPY parameter on either the Create Job Description (CRTJOBD) or Change Job Description (CHGJOBD) command to specify either *DFT or *SYSRPYL in the job description for the target job. You can also do the same thing if you use the SBMRMTCMD command to submit the Change Job (CHGJOB) command to the target system.

  - If you want to specify a default reply message for a specific *inquiry message*

in the job, you can use the Add Reply List Entry (ADDRPYLE) command (on the target system) to add an entry for that message to the system-wide automatic message reply list (SYSRPYL). Then, if INQMSGRPY(*SYSRPYL) is specified in the job description, this default reply can be sent whenever that inquiry message occurs in the job.

WRKDDMF (Work with DDM Files) Command

The Work with DDM Files (WRKDDMF) command allows you to work with existing DDM files from a list display. From the list display, you can change, delete, display, or create DDM files.

For the following displays, it is assumed that you have created DDM files using the Create DDM File (CRTDDMF) command. If you enter the WRKDDMF command and specify library WILSON and file A, the following display is shown:

```
                          Work with DDM Files

Position to . . . . . _____________

Type options, press Enter.
  1=Create DDM file   2=Change DDM file   4=Delete   5=Display details
  6=Print details

                                                                  Remote
Option   Local File              Remote File                      Location
  _      _____________
  _      WILSON/A                 A                                S36

                                                                  Bottom
F3=Exit   F5=Refresh   F9=Print list   F12=Cancel
```

To *create* a DDM file using this display, type a 1 in the option column and type the names of the library and file you want to create, then press the Enter key. For example, type a 1 (Create DDM file) in the option field and WILSON/TEST in the local file column of the top list entry (as shown in the following display), and then press the Enter key. The Create DDM File display is shown.

```
                          Work with DDM Files

Position to . . . . . _____________

Type options, press Enter.
  1=Create DDM file   2=Change DDM file   4=Delete   5=Display details
  6=Print details

                                                                  Remote
Option   Local File              Remote File                      Location
  1      WILSON/TEST_____
  _      WILSON/A                 A                                S36

                                                                  Bottom
F3=Exit   F5=Refresh   F9=Print list   F12=Cancel
```

```
CRTDDMF                  Create DDM File

Type choices, press Enter.

DDM file . . . . . .  TEST_____    Name
  Library . . . . . .   WILSON___   Name, *CURLIB
Remote file
  File . . . . . .  *NONSTD___     Name, *NONSTD
    Library . . . . .  _____       Name, *LIBL, *CURLIB
  Nonstandard file 'name'  . . .  'TESTFILE.TESTLIB'_____
_____
_____
_____
Remote location name . . . . . .  S36____   Name
Text 'description' . . . . . .  *BLANK_____

                                                                  Bottom
F3=Exit   F4=List   F5=Refresh   F10=Additional parameters   F11=Keywords
F12=Cancel   F13=Prompter help
```

On the Create DDM File display, type the required values, and change or use the default values given. See the *APPC Programmer's Guide* for additional information on the CRTDDMF command and its parameters. By pressing F10 (Additional parameters), you can page through the command parameters as they are shown on two displays. By pressing the Page Down key, you are shown these additional parameters:

```
CRTDDMF                Create DDM File

Type choices, press Enter.

                          Additional Parameters

Device
  APPC device description  . . .  *LOC         Name, *LOC
Local location name  . . . . . .  *LOC         Name, *LOC, *NETATR
Mode . . . . . . . . . . . . . .  *NETATR      Name, *NETATR
Remote network identifier  . . .  *LOC         Name, *LOC, *NETATR, *NONE
Access Method
  Remote file attribute  . . . .  *RMTFILE     *RMTFILE, *COMBINED...
  Local access method  . . . . .  _______      *BOTH, *RANDOM, *SEQUENTIAL
Share open data path . . . . . .  *NO          *NO, *YES
Record format level check  . . .  *RMTFILE     *RMTFILE, *NO
Authority  . . . . . . . . . . .  *CHANGE      Name, *CHANGE, *ALL...



                                                                      Bottom
F3=Exit  F4=List  F5=Refresh  F11=Keywords  F12=Cancel  F13=Prompter Help
```

After you have typed in the values, press the Enter key to process the command and return to the Work with DDM Files display.

If you want to *change* a DDM file, type a 2 (Change DDM file) on the Work with DDM Files display next to the file that you want to change, or type the option number in the top list entry of the Options column and specify the local file that you want changed. For example, type a 2 (Change DDM file) in the *Option* column of the local file named WILSON/TEST.

```
                          Work with DDM Files

Position to  . . . . . .  _____________________

Type options, press Enter.
  1=Create DDM file   2=Change DDM file   4=Delete   5=Display details
  6=Print details

                                                                  Remote
Option   Local File              Remote File                      Location
  _      _____________
  _      WILSON/A                A                                S36
  2      WILSON/TEST             TESTFILE.TESTLIB                 S38




                                                                      Bottom
F3=Exit   F5=Refresh   F9=Print list   F12=Cancel
```

Press the Enter key and the Change DDM File display is shown.

For example, if you *only* want to add a text description, type in the description and press the Enter key. But, if you want to make additional changes, press F10 (Additional parameters), and you can page through the command parameters as they are shown on two displays.

```
CHGDDMF                Change DDM File

Type choices, press Enter.

DDM file . . . . . .  .> TEST          Name
  Library  . . . . . .  .>   WILSON     Name, *LIBL, *CURLIB
Remote file
  File . . . . . . . .  .> *NONSTD___    Name, *SAME
    Library  . . . . .  .   _________    Name, *LIBL, *CURLIB
  Nonstandard file 'name'  . .> 'testfile.testlib'_____________
_____________________________________________________________
_____________________________________________________________
_____________________________________________________________
Remote location name . . . . . .> S38          Name, *SAME, *DEVD
Record format level check  . . .> *RMTFILE     *SAME, *RMTFILE, *NO
Text 'description' . . . . . . .> 'TEST VERSION FOR DDM'_______
_____________________________


F3=Exit  F4=List  F5=Refresh  F10=Additional parameters  F11=Keywords
F12=Cancel  F13=Prompter help
```

If you want to change the mode parameter, type in that value, and then press the Enter key.

```
CHGDDMF                Change DDM File

Type choices, press Enter.

                          Additional Parameters
Device
  APPC device description  . . . > *SAME        Name, *SAME, *LOC
Local location name  . . . . . . > *SAME        Name, *SAME, *LOC, *NETATR
Mode . . . . . . . . . . . . . . > S38MODE1     Name, *SAME, *NETATR
Remote network identifier  . . . > *SAME        Name, *SAME, *LOC, *NETATR...
Access method
  Remote file attribute  . . . . > *SAME        *SAME, *RMTFILE, *COMBINED
  Local access method  . . . . .  _________     *BOTH, *RANDOM, *SEQUENTIAL
Share open data path . . . . . . > *SAME        *SAME, *NO, *YES




                                                                      Bottom
F3=Exit  F4=List  F5=Refresh  F11=Keywords  F12=Cancel  F13=Prompter help
```

After you press the Enter key, you return to the Work with DDM Files display.

If you want to *display* the details of a DDM file, type a 5 (Display details) on the Work with DDM Files display next to the file that you want to display, or type the option number in the top list entry of the Options column and specify the local file you want to display. For example, type a 5 (Display details) in the *Option* column and type WILSON/TEST in the *Local File* column of the top list entry.

You can also display the details of a file by using the Display DDM Files (DSPDDMF) command.

```
                          Work with DDM Files

Position to . . . . . _____________

Type options, press Enter.
  1=Create DDM file   2=Change DDM file   4=Delete   5=Display details
  6=Print details

                                                                  Remote
Option   Local File              Remote File                      Location
  5      WILSON/TEST_________
  _      WILSON/A                 A                                S36
  _      WILSON/TEST              TESTFILE.TESTLIB                 S38

                                                                  Bottom
F3=Exit   F5=Refresh   F9=Print list   F12=Cancel
```

Press the Enter key and the Display Details of DDM File display is shown.

```
                    Display Details of DDM File          SYSTEM:  AS400B

Local file:
  File . . . . . :   TEST
    Library . . . . . :     WILSON

Remote file . . . . . :   TESTFILE.TESTLIB


Remote location:
  Remote location . . . . . :   S38
  Device description . . . . . :   *LOC
  Local location . . . . . :   *LOC
  Remote location network ID . . . . . :   *LOC
  Mode . . . . . :   S38MODE1

  Press Enter to continue.
                                                                  More...
  F3=Exit    F12=Cancel
```

Page down to see the second display.

```
                    Display Details of DDM File
                                                  SYSTEM:  AS400B

Access method
  Remote file attribute . . . . :   *RMTFILE
  Local access method . . . . . :

Share open data path . . . . . :   *NO
Check record format level ID . . :   *RMTFILE
Text . . . . . :   TEST VERSION FOR DDM

  Press Enter to continue.
                                                                  Bottom
  F3=Exit    F12=Cancel
```

Press the Enter key to return to the Work with DDM Files display.

In addition to displaying the details of the DDM file, you can *print* the detail information by typing a 6 (Print details) in the *Option* column.

You can also print a list of the DDM files by pressing F9 (Print list).

To *delete* a file or files, type a 4 (Delete) in the *Option* column next to the files you want to delete or in the top list entry and specify the file you want to delete.

```
                          Work with DDM Files

Position to . . . . . _____________

Type options, press Enter.
  1=Create DDM file   2=Change DDM file   4=Delete   5=Display details
  6=Print details

                                                                  Remote
Option   Local File              Remote File                      Location
  _      _________________
  _      WILSON/A                 A                                S36
  4      WILSON/TEST              TESTFILE.TESTLIB                 S38

                                                                  Bottom
F3=Exit   F5=Refresh   F9=Print list   F12=Cancel
```

Press the Enter key. You are shown the Confirm Delete of Files display.

```
                        Confirm Delete of Files

Press Enter to confirm your choices for 4=Delete.
Press F12 to return to change your choices.

                                                                  Remote
Option   Local File              Remote File                      Location
  4      WILSON/TEST              TESTFILE.TESTLIB                 S38

                                                                  Bottom
F12=Cancel
```

Choose one of the actions on the display and then press the Enter key. You return to the Work with DDM Files display.

DDM-Related CL Command Considerations

The following topics describe DDM-related specifics about AS/400 CL commands when they are used with DDM files. These topics discuss running the commands on the source system and do not discuss them being submitted to run on the target system by the Submit Remote Command (SBMRMTCMD) command. These and other commands are organized into various groups later in this chapter. See "DDM-Related CL Command Lists" on page 5-19 for this kind of information.

The following CL command descriptions are arranged in alphabetic order by command name. For complete non-DDM-related information about any of these commands, refer to the *CL Reference* manual.

ALCOBJ (Allocate Object) Command

When the name of a DDM file is specified on the Allocate Object (ALCOBJ) command on the source system, the command allocates the DDM file on the source system and its associated file or file member on a target system. The command places locks on both the DDM file and the remote file in each pair. (These files are locked on both systems to ensure that they are not changed or deleted while the files or members are locked.) One or more pairs of files (DDM files on the source system and remote files on one or more target systems) can be allocated at the same time.

Each DDM file is always locked with a shared-read (*SHRRD) lock. Shared-read is used for the DDM files regardless of the lock types that may have been specified on the command to lock other local files at the same time.

The lock placed on the *remote file* depends on the type of target system:

- When the target is an AS/400 system or a System/38, the resulting locks on the remote file are the same as if the file is a local database file. That is, the AS/400 or the System/38 remote file is also locked with a shared-read lock, and the member (the one specified, or the first one) is locked with the lock type specified on the command.
- When the target is *not* an AS/400 system or a System/38, the remote file is locked with the specified lock type, except that some non-AS/400 target systems may use a stronger lock than was specified on the command. If an ALCOBJ command specifies multiple DDM files, and one or more are on non-AS/400 target systems, those remote files are locked with the lock type specified on the command. If a member name is specified for a remote system that does not support members, the lock request is rejected with an error message, unless the member name is the same as the DDM file name.

Member Names and AS/400 Target Systems: If a member name is specified with the DDM file name on an ALCOBJ command, the member (in the remote file) is locked with the lock type specified on the command. If a member name is also specified in the DDM file itself, the member names on both commands (ALCOBJ and CRTDDMF) must be the same. If they are different, the lock request is rejected and an error message is sent to the user of the program. The remote file containing the member is locked with a shared-read lock regardless of the lock type specified for the member.

If no member name is specified when a DDM file name is specified on an ALCOBJ command for a remote file on an AS/400 system or a System/38, *FIRST is the default, and the target system attempts to locate and lock the first member in the remote file, the same as if it had been specified by name. If a remote file has no members, the lock request is rejected with an error message.

Locking Multiple DDM Files: One ALCOBJ command can be used to specify multiple DDM files that are associated with remote files located on multiple target systems. If it is not possible to lock all the files on all the systems, none are locked.

Command Completion Time: When DDM-related files are being allocated, a longer time will be required for the command to complete because of the additional time required for communications to occur between the source and target systems. You should not, however, increase the wait time specified in the WAIT parameter on the Allocate Object (ALCOBJ) command; commu-

nications time and the WAIT parameter value have no relationship with each other.

Note: If the DLTF command is used to delete the remote file without first releasing (using the DLCOBJ command) the locks obtained by the ALCOBJ command, the DDM conversation is not reclaimed until the source job has ended.

CHGJOB (Change Job) Command

The Change Job (CHGJOB) command can be used to change the DDMCNV parameter, which controls whether advanced program-to-program communications (APPC) or PC Support/400 conversations allocated for DDM use are to be kept active or automatically dropped when they are not in use by a job. The new value goes into effect immediately for the specified job.

To display the current value of the DDMCNV job attribute, use the Work with Job (WRKJOB) command (described later).

See "DDMCNV Parameter Considerations" on page 5-18 for a description of this parameter's values.

CHGLF (Change Logical File) Command

The Change Logical File (CHGLF) command can be used to change files on the source and target systems through the SYSTEM parameter. Consider the following items when using the SYSTEM parameter values:

- When you specify *LCL, the logical file is changed on the local system.
- When you specify *RMT, the logical file is changed on the remote system. You must specify a DDM file on the FILE parameter.
- When you specify *FILETYPE, a remote file is changed if a DDM file has been specified on the FILE parameter. If a DDM file has not been specified, a local logical file is changed.

Consider the following items when using this command with DDM:

- The FILE parameter is the name of the DDM file that represents the remote logical file being changed. The remote file specified on the DDM file is the logical file that is changed on the remote system (which is also specified in the DDM file).
- For a target system other than an AS/400 system:
  - All parameters except TEXT are ignored.
  - It is not verified that the remote file is a logical file.

CHGPF (Change Physical File) Command

The Change Physical File (CHGPF) command can be used to change files on the source and target systems through the SYSTEM parameter. Consider the following items when using the SYSTEM parameter values:

- When you specify *LCL, the physical file is changed on the local system.
- When you specify *RMT, the physical file is changed on the remote system. You must specify a DDM file on the FILE parameter.
- When you specify *FILETYPE, if a DDM file has been specified on the FILE parameter, a remote file is changed. If a DDM file has not been specified, a local physical file is changed.

Consider the following items when using this command with DDM:

- The FILE parameter is the name of the DDM file that represents the remote physical file being changed. The remote file specified in the DDM file is the physical file that is changed on the remote system (which is also specified in the DDM file).
- For a target system other than an AS/400 system:
  - All parameters except EXPDATE, SIZE, and TEXT are ignored.
  - It is not verified that the remote file is a physical file.

CHGSRCPF (Change Source Physical File) Command

The Change Source Physical File (CHGSRCPF) command can be used to change files on the source and target systems through the SYSTEM parameter. Consider the following items when using the SYSTEM parameter values:

- When you specify *LCL, the source physical file is changed on the local system.
- When you specify *RMT, the source physical file is changed on the remote system. You must specify a DDM file on the FILE parameter.
- When you specify *FILETYPE, if a DDM file has been specified on the FILE parameter, a remote file is changed. If a DDM file has not been specified, a local source physical file is changed.

Consider the following items when using this command with DDM:

- The FILE parameter is the name of the DDM file that represents the remote source physical file being changed. The remote file specified in the DDM file is the source physical file that is changed on the remote system (which is also specified in the DDM file).
- For a target system other than an AS/400 system, the CHGSRCPF command cannot be used to change files.

CLRPFM (Clear Physical File Member) Command

The Clear Physical File Member (CLRPFM) command can be used with DDM to clear all the records either from a physical file member on a target AS/400 system or from a file on a non-AS/400 target system. The command works the same way as it does for local files (clearing all data records and deleted records).

CPYF (Copy File), CPYSRCF (Copy Source File), and CPYFRMQRYF (Copy from Query File) Commands

The Copy File (CPYF), Copy Source File (CPYSRCF), and Copy from Query File (CPYFRMQRYF) commands can be used to copy data or source between files on source and target systems. You can copy local database or device files from (or to) remote database files regardless of whether the target system is an AS/400 system. Remote files can also be copied to remote files.

Consider the following items when using this command with DDM:

- A DDM file can be specified on the FROMFILE and the TOFILE parameters for the CPYF and CPYSRCF commands.

  Note: For the Copy from Query File (CPYFRMQRYF), Copy from Diskette (CPYFRMDKT) and Copy from Tape (CPYFRMTAP) commands, a DDM file name can be specified only on the TOFILE parameter; for the Copy to Diskette (CPYTODKT) and Copy to Tape (CPYTOTAP) commands, a DDM file name can be specified only on the FROMFILE parameter.
- If the target system is *not* an AS/400 system or a System/38:
  - When a file on the local AS/400 system is copied to a remote file (or vice versa), FMTOPT(*NOCHK) is usually required.
  - When a *source* file on the local AS/400 system is copied to a remote file (or vice versa), FMTOPT(*CVTSRC) *must* be specified.
- If data is copied to a target System/36 file that has alternative indexes built over it, the CPYF command cannot specify MBROPT(*REPLACE). In this case, the CPYF command attempts to clear the remote file, but it fails because of the alternative indexes.
- When a delete-capable file on the local AS/400 remote file is copied to a non-delete capable file, you must specify COMPRESS(*YES), or an error message is sent and the job ends.
- If the remote file name on a DDM file specifies a member name, the member name specified for that file on the CPYF command must be the same as the member name on the remote file name on the DDM file. In addition, the Override Database File (OVRDBF) command cannot specify a member name that is different from the member name on the remote file name on the DDM file.
- If a DDM file does not specify a member name and if the OVRDBF command specifies a member name for the file, the CPYF command uses the member name specified on the OVRDBF command.
- If the TOFILE parameter is a DDM file that refers to a file that does not exist, CPYF creates the file. Following are special con-

siderations for remote files created with the CPYF command:

- If the target system is an AS/400 system or a System/38, the user profile for the target DDM job must be authorized to the CRTPF command on the target system.
- If the target system is not an AS/400 system, the FROMFILE parameter cannot be a source file.
- If the target system is a System/38, the TOMBR parameter must be the same as the remote file's name or *FIRST for the copy to be successful. The copy creates a member with the same name as the remote file's name.
- If the target system is a System/36 and the FROMFILE parameter is a logical file or a physical file with SIZE(*NOMAX), a file with 8 388 607 records is created.
- If the target system is a System/36, the TOMBR parameter must be the DDM file's name or *FIRST for the copy to be successful. For DDM access to the remote file, the file appears to have a member with the same name as the DDM file.
- For an AS/400 system target, the TOFILE parameter has all the attributes of the FROMFILE parameter except those described in the *Data Management Guide*.
- For non-AS/400 system targets, those attributes on the CRTPF command that are also ignored for non-AS/400 systems are ignored when CPYF creates the file.
- If the target system is a System/38 and the FROMFILE parameter is a direct file that does not allow deleted records, an attempt is made to copy the records after the last record for the file at its maximum size. The system operator on the System/38 tells the system to either add the records or cancel the copy.

CRTLF (Create Logical File) Command

The Create Logical File (CRTLF) command can be used to create files on the source and target systems through the SYSTEM parameter. Consider the following items when using the SYSTEM parameter values:

- When you specify *LCL, the file is created on the local system.
- When you specify *RMT, the file is created on the remote system. You must specify a DDM file on the FILE parameter.
- When you specify *FILETYPE, if a DDM file has been specified on the FILE parameter, a remote file is created. If a DDM file has not been specified, a local file is created.

Consider the following items when using this command with DDM:

- The parameter FILE is the name of the DDM file that represents the remote logical file being created. The remote file specified in the DDM file is the logical file that is created on the remote system (which is also specified in the DDM file).
- The OPTION and GENLVL parameters have no effect on the remote command sent.
- The files specified on the PFILE or JFILE keywords in the DDS for the logical file must be at the same system location as the logical file being created.
- For a target system other than an AS/400 system:
  - The format name is ignored.
  - Only the value of *ALL is supported for the DTAMBRS parameter.
  - These parameters are ignored: MBR, AUT, LVLCHK, MAINT, RECOVER, FRCACCPTH, UNIT, FRCRATIO, WAITFILE, WAITRCD, and SHARE.

    Note: For System/38 targets, the SBMRMTCMD command can be used to change these attributes.
  - Only the value of *NONE is supported for the FMTSLR parameter.
  - FILETYPE must be *DATA.

- If a member name is specified, it must match the DDM file name.

- For an AS/400 target system:
  - All parameters of the CRTLF command are supported with one restriction: authorization lists are not allowed for the AUT (public authority) parameter. DDM cannot guarantee the existence of the authorization list on the target system or that the same user IDs are in the list if it does exist. The public authority is changed to *EXCLUDE when you use an authorization list as a value for the AUT parameter of the CRTLF command.
  - The file names specified in the DTAMBRS parameter must be the names of the DDM files that represent the remote based-on physical files. If a member name was specified as part of the remote file name of the DDM file, then only that member name can be specified. The member names must be the actual remote file member names.

CRTPF (Create Physical File) Command

The Create Physical File (CRTPF) command can be used to create files on the source and target systems through the SYSTEM parameter. Consider the following items when using the SYSTEM parameter values:

- When you specify *LCL, the file is created on the local system.
- When you specify *RMT, the file is created on the remote system. You must specify a DDM file on the FILE parameter.
- When you specify *FILETYPE, if a DDM file has been specified on the FILE parameter, a remote file is created. If a DDM file has not been specified, a local file is created.

Consider the following items when using this command with DDM:

- The FILE parameter is the name of the DDM file that represents the remote file being created. The remote file specified in the DDM file is the file that is created on the remote system (which is also specified in the DDM file).
- The OPTION and GENLVL parameters create the same results as for local processing. These parameters have no effect on the remote command sent.
- For a target system other than an AS/400 system:
  - The format name is ignored.
  - These parameters are ignored: MBR, MAXMBRS, MAINT, RECOVER, FRCACCPTH, CONTIG, UNIT, FRCRATIO, WAITFILE, WAITRCD, SHARE, DLTPCT, REUSEDLT, LVLCHK, and AUT.

    Note: For System/38 targets, the SBMRMTCMD command can be used to change these attributes.
  - FILETYPE must be *DATA.
  - All other parameters are supported.
  - If a member name is specified, it must match the DDM file name.
- For AS/400 target systems:
  - All parameters of the CRTPF command are supported with one restriction: authorization lists are not allowed for the AUT (public authority) parameter. DDM cannot guarantee the existence of the authorization list on the target system or that the same user IDs are in the list if it does exist. The public authority is changed to *EXCLUDE when you use an authorization list as a value for the AUT parameter of the CRTPF command.

CRTSRCPF (Create Source Physical File) Command

The Create Source Physical File (CRTSRCPF) command can be used to create files on the AS/400 source and target systems through the SYSTEM parameter. Consider the following items when using the SYSTEM parameter values:

- When you specify *LCL, the file is created on the local system.
- When you specify *RMT, the file is created on the remote system. You must specify a DDM file on the FILE parameter.
- When you specify *FILETYPE, if a DDM file has been specified on the FILE parameter, a

remote file is created. If a DDM file has not been specified, a local file is created.

Consider the following items when using this command with DDM:

- The FILE parameter is the name of the DDM file that represents the remote file being created. The remote file specified in the DDM file is the file that is created on the remote system (which is also specified in the DDM file).
- The OPTION and GENLVL parameters create the same results as for local processing. These parameters have no effect on the remote command sent.

All parameters of CRTSRCPF are supported with one restriction: authorization lists are not allowed for the AUT (public authority) parameter. DDM cannot guarantee the existence of the authorization list on the target system or that the same user IDs are in the list if it does exist. The public authority is changed to *EXCLUDE when you use an authorization list as a value for the AUT parameter of the CRTSRCPF command.

DLCOBJ (Deallocate Object) Command

When the name of a DDM file is specified on the Deallocate Object (DLCOBJ) command on the source system, the command deallocates the DDM file on the source system and its associated file or file member on a target system. The command releases the locks that were placed on the paired files on both the source and target systems by the Allocate Object (ALCOBJ) command. One or more pairs of files (DDM files on the source system and remote files on one or more target systems) can be deallocated at the same time.

Member Names and AS/400 Target Systems: All of the information previously discussed in the ALCOBJ command description regarding member names applies to the DLCOBJ command as well. Refer to the ALCOBJ command description for the details.

Unlocking Multiple DDM Files: One DLCOBJ command can be used to specify multiple DDM files that are associated with remote files that may be located on multiple target systems. In most cases, the command attempts to release as many of the specified locks as possible. For example:

- If one of the DDM files specified on the DLCOBJ command refers to a remote file that is not a database file, that lock is not released; but the specified locks on the remote files associated with the other DDM files specified are released if, of course, they are valid.
- If a user tries to release a lock that he did not place on a file by a previous ALCOBJ command, that part of the request is rejected and an informational message is returned to the user.

DLTF (Delete File) Command

The Delete File (DLTF) command can be used to delete files on the source and target systems. The following items should be considered when using the SYSTEM parameter values:

- When you specify *LCL, only local files are deleted. This may include DDM files.
- When you specify *RMT, the file is deleted on the remote system. You must specify a DDM file on the FILE parameter. If a generic name is specified, the remote files corresponding to any DDM files matching the generic name are deleted. (The local DDM files are not deleted.)
- When you specify *FILETYPE, if a DDM file has been specified, the remote file is deleted. If a DDM file has not been specified, the local file is deleted. When you specify generic names, local non-DDM files are deleted first. Remote files for any DDM files matching the generic name are then deleted. Local DDM files are not deleted.

Notes:

1. Structured Query Language/400 (SQL/400*) DROP TABLE and DROP VIEW statements work only on local files.
2. If the DLTF command is used to delete the remote file without first releasing (using the DLCOBJ command) the locks obtained by the ALCOBJ command, the DDM conversation is not reclaimed until the source job has ended.

DSPFD (Display File Description) Command

The Display File Description (DSPFD) command can be used to display (on the source system) the attributes of the DDM file on the source system, the remote file on the target system, or both the DDM file and the remote file. As with local files, the attributes of multiple DDM and/or remote files can be displayed by the same command.

Note: Although this discussion mentions only one target system, the files for multiple target systems can be displayed at the same time.

The SYSTEM parameter determines which group of attributes is displayed.

- To display the attributes of DDM files, which are *local* files, the SYSTEM parameter must specify *LCL (the default). If SYSTEM(*LCL) is specified:
  - The FILEATR parameter must either specify *DDM (to display DDM file attributes only) or default to *ALL (to display all file types, including DDM files). The same kind of information is displayed for DDM files (which are on the local system) as for any other types of files on the local system.
  - If FILEATR(*DDM) is specified and the OUTFILE parameter specifies a file name, only local DDM file information is given.
- To display the attributes of remote files, the SYSTEM parameter must specify *RMT. If SYSTEM(*RMT) is specified:
  - The FILEATR parameter must specify *ALL, *PHY, or *LGL.
  - The type of information displayed for remote files depends on what type of target system the files are on. If the target is an AS/400 system or a System/38, the same type of information displayed for local files on an AS/400 system or a System/38 can be displayed. If the target is *not* an AS/400 system or a System/38, all the information that can be obtained through that system's implementation of the DDM architecture that is compatible with the AS/400 system's implementation is displayed.
- To display the attributes of both DDM and remote files, the SYSTEM parameter must specify *ALL.

DSPFFD (Display File Field Description) Command

The Display File Field Description (DSPFFD) command can be used to display the file, record format, and field attributes of a remote file. To display the remote file attributes, however, you must enter the name of the DDM file associated with the remote file, not the name of the remote file.

Note: Because the DDM file has no field attributes, the DSPFFD command cannot specify SYSTEM(*LCL) to display local DDM file information.

If *ALL or a generic file name is specified on the FILE parameter, the DSPFFD command can also display information about a group of both local files and remote files, or just a group of local files. In this case, the SYSTEM parameter determines which are displayed.

- To display the attributes of *local non-DDM* files only, the SYSTEM parameter need not be specified because *LCL is the default.
- To display the attributes of *remote* files, the SYSTEM parameter must specify *RMT. If SYSTEM(*RMT) is specified, the field and record format information displayed for remote files depends on what type of target system the files are on.
  - If the target is an AS/400 system or a System/38, the same information displayed for local files on an AS/400 system is displayed.
  - If the target is neither an AS/400 system nor a System/38, fields are F*nnnnn* or K*nnnnn* (where *nnnnn* is some number), based on whether the file is a keyed file or not. The record format name is the DDM file name.
- To display the attributes of *both local non-DDM files and remote files*, the SYSTEM parameter must specify *ALL. Only remote physical and logical files can be displayed.

OVRDBF (Override with Database File) Command

The Override with Database File (OVRDBF) command can be used with DDM to override (replace) a local database file named in the program with a DDM file; the DDM file causes the associated remote file to be used by the program instead of the local database file.

If a DDM file is specified on the TOFILE parameter and if other parameters are specified that change a file's attributes, the result is that the remote file actually used by the program is used with its attributes changed by the parameter values specified on the OVRDBF command.

If the target system is an AS/400 system or a System/38, existing programs that use the OVRDBF command to access remote files work the same as when they access local files. All the OVRDBF parameters are processed the same on source and target AS/400 systems.

Because of the way data management handles overrides, the user must be careful when specifying a member name on the Override with Database File (OVRDBF) command. If a member name is specified, data management first searches for a local database file containing the member specified, before looking for a DDM file.

For purposes of this example, you can assume the following:

- DDM file CUST021 is in library NYCLIB.
- Database file CUST021 is in library CUBSLIB.

NYCLIB is listed before CUBSLIB in the user's library list. CUBSLIB/CUST021 contains member NO1. The remote file pointed to by the DDM file contains member NO1. If the override is used:

```
OVRDBF FILE(CUST021) MBR(NO1)
```

Data management finds the database file CUBSLIB/CUST021 instead of the DDM file NYCLIB/CUST021.

To avoid this, you can do one of the following:

- Qualify the TOFILE on the override:

  ```
  OVRDBF FILE(CUST021) TOFILE(NYCLIB/CUST021) MBR(NO1)
  ```

- Remove the library containing the database file from the library list:

  ```
  RMVLIBLE LIB(CUBSLIB)
  ```

- Remove the override and change the remote file name in the DDM file to contain the member name:

  ```
  CHGDDMF FILE(NYCLIB/CUST021)
     RMTFILE(*NONSTD 'XYZ/CUSTMAST(NO1)')
  ```

If end-of-file delay (EOFDLY) is used, it is recommended to end a job with an end-of-file record because if the source job gets canceled, the target job does not get notified. The user must also end the target job.

If the target system is neither an AS/400 system nor a System/38:

- The following parameters are still valid: TOFILE, POSITION, RCDDMTLCK, WAITFILE, WAITRCD, LVLCHK, EXPCHK, INHWRT, SECURE, SHARE, and SEQONLY.
  - The TOFILE parameter is always processed on the source system. When a DDM file name is specified on this parameter, the program uses the associated remote file instead of the local database file specified in the program.
  - The RCDDMTLCK parameter, if specified, is valid only if both of the following are true of the remote file used: only one type of lock condition can be requested for the remote file, and the record format name in the remote file must be the same as the name of the DDM file.
  - The WAITFILE and WAITRCD parameters have no effect on remote file processing.
- The MBR parameter causes an error if it is specified with a member name that is different than the name of the file containing the member.
- The FRCRATIO and NBRRCDS parameters, if specified, are ignored.
- The FMTSLR parameter, if specified, causes an error when the file being opened is a DDM file.
- The SEQONLY parameter causes records to be blocked on the source side. Records may be lost if the source job is canceled before a block is full.

For examples of how file overrides are applied in DDM, see "Additional Considerations" on page 5-5 for the SBMRMTCMD command description, and see "Examples of Accessing

Remote Members (AS/400 System Only)" on page 6-5.

RCLRSC (Reclaim Resources) Command

The Reclaim Resources (RCLRSC) command, like the Reclaim DDM Conversations (RCLDDMCNV) command, can be used to reclaim all DDM conversations that currently have no users in the job, as defined under "Controlling DDM Conversations" on page 6-8. (This can be done even if the DDMCNV job attribute is *KEEP.) The RCLRSC command, however, first attempts to close any unused files for the appropriate recursion levels, as it would for local files. This action may result in some conversations allocated to DDM being unavailable for the job. For example, if a DDM file is opened using the Open Database File (OPNDBF) command, the RCLRSC command closes the file and reclaims the conversation.

After the files are closed, any unused DDM conversations are dropped. Whether or not a conversation can be reclaimed is not affected by the recursion level in which the RCLRSC command is issued.

RNMOBJ (Rename Object) Command

The Rename Object (RNMOBJ) command can be used to rename a remote file. The following items should be considered when using the SYSTEM parameter values:

- When you specify *LCL, local objects are renamed. This may include DDM files.
- When you specify *RMT, this value applies only to OBJTYPE(*FILE). The DDM file containing the remote file to be renamed is specified on the OBJ parameter.

  The DDM file containing the new name for the remote file is specified on the NEWOBJ parameter. Both DDM files must already exist in the same library (on the source system). The two DDM files must refer to the same target systems and contain the same remote location information. Neither the two local DDM files nor the RMTFILE names in the two DDM files are changed. Specify *LCL to rename the DDM file or use the Change DDM File (CHGDDMF) command to change the RMTFILE name in a DDM file.
- When you specify *FILETYPE, this value applies only to OBJTYPE(*FILE). If the file specified in the OBJ parameter is a DDM file, the rules when specifying *RMT apply. If the file is not a DDM file, the rules when specifying *LCL apply.

When renaming remote files for AS/400 and System/38 targets, if library names have been specified in the RMTFILE parameter for the two DDM files, the library names must be the same but the file names must be different.

WRKJOB (Work with Job) Command

The Work with Job (WRKJOB) command can be used to display two DDM-related items:

- The DDMCNV job attribute for the source job. See "DDMCNV Parameter Considerations" on page 5-18 for a description of the values for this attribute.
- The object lock requests (held locks and pending locks) for DDM files that are being used in the source system job. These are shown by choosing option 12 (Work with locks, if active) from the Work with Job menu.

The Job Locks display shows only the locks held for the local DDM files; locks for remote files are not shown. Also, because DDM files do not have members, none are indicated on this display nor on the Member Lock display.

An AS/400 system does not display any locks for remote files; locks for the remote file, its members, or its records cannot be displayed by the source system. However, these remote locks can be displayed using pass-through.

The lock condition shown for DDM files is always shared read (*SHRRD) regardless of the lock conditions used for their associated remote files or members.

WRKOBJLCK (Work with Object Lock) Command

The Work with Object Lock (WRKOBJLCK) command can be used to display the object lock requests (held locks and pending locks) for DDM files. This command displays only the locks held

for the local DDM files, not locks held for the associated remote files.

An AS/400 system does not display any locks for remote files; locks for the remote file, its members, or its records cannot be displayed by the source system.

The lock condition shown for DDM files is always shared read (*SHRRD) regardless of the lock conditions used for their associated remote files or members.

DDM-Related CL Parameter Considerations

The following parameter considerations apply to DDM-related CL commands:

- The **DDMACC parameter** controls how an AS/400 system, as a target system, handles DDM requests from other systems.
- The **DDMCNV parameter** controls, in a source system job, whether unused DDM conversations are to be kept active or automatically dropped.
- For *commands* that cannot specify a DDM file name, see "Commands Not Supporting DDM" on page 5-22.

  Note: The Create DDM File (CRTDDMF) command can be used to create a DDM file. The other create file commands (such as CRTPF or CRTxxxF) cannot be used to create a DDM file.
- The **OUTFILE parameter** can specify a DDM file only if the remote system is an AS/400 system or a System/38 and only if the file already exists on the remote AS/400 system or System/38.

DDMACC Parameter Considerations

The DDMACC parameter is used on the Change Network Attributes (CHGNETA), Display Network Attributes (DSPNETA), and Retrieve Network Attributes (RTVNETA) commands. The value of this system-level parameter determines whether this AS/400 system can accept DDM requests from other systems. The values for this parameter are discussed as part of target system security under "DDM Network Attribute (DDMACC Parameter)" on page 4-3.

DDMCNV Parameter Considerations

The DDMCNV parameter is a job-related parameter that controls whether advanced program-to-program communications (APPC) or AS/400 conversations in the job that are allocated for DDM use (that is, DDM conversations) are to be automatically dropped or kept active for the source job. The default is to keep the conversation active.

This parameter can drop a conversation when it has no active users. The conversation is unused when:

1. All the DDM files and remote files used in the conversation are closed and unlocked (deallocated).
2. No other DDM-related functions (like the Submit Remote Command [SBMRMTCMD] command or the Display File Description [DSPFD] command to access the target system) are being done.
3. No DDM-related function has been interrupted (by a break program, for example) while running.

For other ways that conversations are normally dropped, or are explicitly dropped by another CL command, see "Controlling DDM Conversations" on page 6-8.

The DDMCNV parameter values are:

***KEEP** Specifies that once each DDM conversation is started for the source job it is kept active at the completion of a source program request, and it waits for another request to be received from the same or another program running in the job. This is the default value.

***DROP** Specifies that each DDM conversation started in the source job is to be automatically dropped when both of the following are true: no request from the source system program(s) running in the job is being processed in the conversation, and all the DDM files are closed and all objects that were allocated are now deallocated.

The DDMCNV parameter is changed by the Change Job (CHGJOB) command and is displayed by the Work with Job (WRKJOB) command. Also, if the Retrieve Job Attributes (RTVJOBA) command is used, you can get the value of this parameter and use it within a CL program.

OUTFILE Parameter Considerations

The OUTFILE parameter is used on such commands as the Display File Description (DSPFD), the Display File Field Description (DSPFFD), the Display Object Description (DSPOBJD), and the Create Auto Report Program (CRTRPTPGM). The parameter identifies a database file into which output data created by the command is stored. When the name of a DDM file is specified on the OUTFILE parameter of these commands, two restrictions apply:

- The remote system must be an AS/400 system or a System/38. This is necessary to ensure that the associated remote file has the proper format for the output data.
- The remote file associated with the DDM file must already exist. If the remote file does not exist, a message is returned to the user indicating that the remote file must exist before the function can be performed.

If the remote file named on the OUTFILE parameter does exist and is on an AS/400 system or a System/38, the file will be checked for three conditions before it can be used as an output database file to store displayed output:

- The remote file must be a physical file.
- The remote file must not be a model outfile. That is, it cannot be one of the model output files provided with OS/400 which has the required format, but no data.
- The record format name in the remote file must match the model outfile record format name. (This condition requires that the remote system be an AS/400 system or a System/38.)

If all of these conditions are met, the remote file member is cleared. (Outfile members *must* be cleared before they can be used again.) If the remote file member does not exist, it is created and the output is stored in it.

DDM-Related CL Command Lists

The control language (CL) commands that have a specific relationship with DDM are grouped in charts in this section to show: the command functions that are available with DDM, those having common limitations when used with DDM, and those that cannot be used with DDM.

Notes:

1. Not *all* of the CL commands on an AS/400 system are shown in this section. Only those that are intended (or recommended) by IBM for use with DDM or those specifically *not* intended for DDM use are shown. The intended use could be either for commands that are run on the source system to affect a remote file on the target system, or for commands that are submitted to the target system via the Submit Remote Command (SBMRMTCMD) command to run there.
2. Some of these commands appear in more than one of the following charts.
3. For a list of all the CL commands that are likely to be used with DDM, see Appendix B.

The charts in this section show:

- Commands affecting only the DDM file:

  Object-oriented commands that can be used with DDM files, but do not affect the associated remote files. The Create DDM File (CRTDDMF), Change DDM File (CHGDDMF), and Reclaim DDM Conversations (RCLDDMCNV) commands are included in this group.

- Commands affecting both the DDM file and the remote file:
  - File management commands that require that the target system be another AS/400 system or a System/38. The SBMRMTCMD command is included in this group.
  - Member-related commands that can be used in some way on remote files.
  - Source file commands that can operate on source files while DDM is being used.

  These commands, normally used for processing local files, can (transparently to the programs) process remote files when one of

their parameters specifies the name of a DDM file.

- Commands that cannot be used with DDM.

Many of these commands, when limited as shown in the charts, can still be submitted by the SBMRMTCMD command to a target system (an AS/400 system or a System/38 only) to run, but it may not be useful to do so. Refer to Appendix B for additional information about these DDM-related commands. Shown, for example, are all the CL commands that can produce meaningful results on the target system when they are submitted on the SBMRMTCMD command.

Object-Oriented Commands

The DDM file object on the source AS/400 system can be accessed by the following object-oriented CL commands. These commands work with DDM files as they normally do with any other files on the local system. Some of these commands can operate on more than one object, and one or more of them could be DDM files if, for example, a generic file name is specified.

Except as noted in the chart, these commands have no effect on the remote file associated with the DDM file; that is, no reference is made over a communications line to the target system when one of these commands specifies a DDM file.

However, if you do want one of these commands to operate on a remote file (instead of the DDM file), you can use the Submit Remote Command (SBMRMTCMD) command to submit the command to run on the target system, if it is an AS/400 system or a System/38. The results of running the submitted command, in this case, are not sent back to the source system, except for some indication to the source system user (normally a message) about whether or not the function was performed successfully.

Command Name	Descriptive Name
CHGDDMF	Change DDM File
CHGLF[1,2,3,4]	Change Logical File
CHGOBJOWN	Change Object Owner
CHGPF[1,2,3,4]	Change Physical File
CHGSRCPF[1,2,3,4]	Change Source Physical File
CHKOBJ	Check Object
CRTDDMF	Create DDM File
CRTDUPOBJ	Create Duplicate Object
CRTLF[1,2,3]	Create Logical File
CRTPF[1,2,3]	Create Physical File
CRTSRCPF[1,2,3]	Create Source Physical File
DLTF[1,2,3]	Delete File
DMPOBJ	Dump Object
DMPSYSOBJ	Dump System Object
DSPFD[1,2,3]	Display File Description
DSPFFD[1,2,3]	Display File Field Description
DSPOBJAUT	Display Object Authority
DSPOBJD	Display Object Description
GRTOBJAUT	Grant Object Authority
MOVOBJ	Move Object
RCLDDMCNV	Reclaim DDM Conversations
RNMOBJ[1,2,3]	Rename Object
RSTLIB	Restore Library
RSTOBJ	Restore Object
RVKOBJAUT	Revoke Object Authority
SAVCHGOBJ	Save Changed Object
SAVLIB	Save Library
SAVOBJ	Save Object
WRKJOB[5]	Work with Job
WRKOBJLCK[5]	Work with Object Lock

Notes:

1 When run on the source system, this command does not refer to the remote file when SYSTEM(*LCL) is used.

2 The remote operation is performed if SYSTEM(*RMT) is specified, or if SYSTEM(*FILETYPE) is specified and the file is a DDM file.

3 Because DDM file names can be specified on these commands, the SBMRMTCMD command is not needed to perform these functions on a target AS/400 system or a target System/38.

4 The target must be an AS/400 system at release 3.0 and above or support Level 2.0 of DDM architecture.

5 When run on the source system, this command displays any locks on the DDM file, not on the remote file.

Target AS/400-Required File Management Commands

The following CL commands can be used only when the target system is another AS/400 system or System/38:

Command Name	Descriptive Name
ADDLFM[1]	Add Logical File Member
ADDPFM	Add Physical File Member
CHGLFM	Change Logical File Member
CHGPFM	Change Physical File Member
CPYSRCF	Copy Source File
INZPFM	Initialize Physical File Member
OPNQRYF	Open Query File
RGZPFM	Reorganize Physical File Member
RMVM	Remove Member
RNMM	Rename Member
SBMRMTCMD	Submit Remote Command

Note:

[1] The target system must be an AS/400 system.

Because DDM file names can be specified on these commands, the Submit Remote Command (SBMRTCMD) command is not needed to perform these functions on a target AS/400 system or a target System/38.

Member-Related Commands

Database file operations that apply to a member can be used by DDM. When the name of a DDM file is specified on any of the following CL commands, OS/400 DDM accesses the remote file (and member) referred to by the DDM file. However, as indicated in the chart, some of these commands are valid only when the remote file is on an AS/400 system or a System/38.

Command Name	Descriptive Name
ADDPFM[1]	Add Physical File Member
ADDLFM[6]	Add Logical File Member
ALCOBJ	Allocate Object
CHGLFM[1]	Change Logical File Member
CHGPFM[1]	Change Physical File Member
CLOF	Close File
CLRPFM	Clear Physical File Member
CPYF[2]	Copy File
CPYFRMDKT	Copy From Diskette
CPYFRMTAP	Copy From Tape
CPYSPLF	Copy Spooled File
CPYSRCF[1]	Copy Source File
CPYTODKT	Copy To Diskette
CPYTOTAP	Copy To Tape
DCLF	Declare File
DLCOBJ	Deallocate Object
DSPFD[3]	Display File Description
DSPFFD[3]	Display File Field Description
DSPPFM	Display Physical File Member
INZPFM[1]	Initialize Physical File Member
OPNDBF[4]	Open Database File
OPNQRYF[1]	Open Query File
OVRDBF[5]	Override Database File
POSDBF	Position Database File
RCVF	Receive File
RCVNETF	Receive Network File
RGZPFM[1]	Reorganize Physical File Member
RMVM[1]	Remove Member
RNMM[1]	Rename Member
SNDNETF	Send Network File

Notes:

[1] The target system must be an AS/400 system or a System/38.

[2] For other DDM-related considerations about this command, see "CPYF (Copy File), CPYSRCF (Copy Source File), and CPYFRMQRYF (Copy from Query File) Commands" on page 5-11.

[3] These commands display remote file information if the SYSTEM parameter specifies *RMT or *ALL.

[4] For information on commitment control, see "Commitment Control Support" on page 2-4.

[5] This command does not access the remote file.

[6] The target system must be an AS/400 system.

The Submit Remote Command (SBMRMTCMD) command can also be used to submit some of the commands to a target system, but only if it is an AS/400 system or a System/38.

The Send Network File (SNDNETF) and Receive Network File (RCVNETF) commands, whenever possible, should run on the system on which the data exists, rather than using a DDM file to

access the remote file. For more information, see "Using Object Distribution" on page 6-10.

Commands Not Supporting DDM

The following CL commands are not supported for DDM files. However, useful results for some of them may be produced on a target AS/400 system or a System/38 using DDM if they are submitted on the Submit Remote Command (SBMRMTCMD) command to run on the target system.

Command Name	Descriptive Name
DSNFMT	Design Format
DSPCHT	Display Chart
DSPDBR	Display Database Relations
DSPRCDLCK	Display Record Locks
MNGDEVTBL	Manage Device Table
MNGPGMTBL	Manage Program Table
MNGUSRTBL	Manage User Table
RTVQRYSRC	Retrieve Query Source
SBMFNCJOB	Submit Finance Job

Source File Commands

If the target system is an AS/400 system or a System/38, the following CL commands can support a DDM file as a source file (on the SRCFILE parameter). If the target system is not an AS/400 system or a System/38, a DDM file name should not be specified on the SRCFILE parameter, because the remote file is neither an AS/400 system nor a System/38 source file.

These commands can also be affected by file overrides (via the Override with Database File [OVRDBF] command).

Note: These commands cannot run on the source system to create a file on any target system; they can, however, be submitted to run on the target system using the Submit Remote Command (SBMRMTCMD) command, if the target is an AS/400 system or a System/38.

Command Name	Descriptive Name
CRTBASPGM	Create BASIC Program
CRTBSCF[1]	Create BSC File
CRTCBLPGM	Create COBOL Program
CRTCLPGM	Create CL Program
CRTCMD	Create Command
CRTCMNF[1]	Create Communications File
CRTCPGM	Create C Program
CRTDSPF	Create Display File
CRTICFF	Create Intersystem Communications Function File[1]
CRTMXDF[2]	Create Mixed File
CRTPLIPGM	Create PL/I Program
CRTPRTF	Create Printer File
CRTPRTIMG	Create Print Image
CRTRPGPGM	Create RPG Program
CRTRPTPGM	Create Auto Report Program
CRTTBL	Create Table
FMTDTA	Format Data
STRBAS	Start BASIC
STRBASPRC	Start BASIC Procedure

Notes:

1 CRTICFF is valid on an AS/400 system. CRTCMNF, CRTBSCF, and CRTMXDF commands are valid either on System/38 or System/38 environment on an AS/400 system.

2 If used with the SBMRMTCMD command, the target must be a System/38.

Data Description Specifications (DDS) Considerations

DDS, which is used to externally describe the fields and record formats, can also be used with DDM to describe the file and record formats of a remote file.

AS/400 Target Considerations

As with any database file, DDS may or may not be used to externally describe the attributes of the remote file when it is created on the remote AS/400 system. If DDS is used, then the source system program uses those attributes when it accesses the remote file (via the DDM file). If DDS is not used, then the file's attributes must be described in the program.

When a source system program that accesses a file on a target AS/400 system is compiled (or recompiled), the existing DDM file is used to

establish communications with the target system, and the remote file is actually accessed during compilation to extract its file and record attributes. Whether or not DDS is used to describe the file, level check identifiers are created during compilation and are included in the compiled program. These values are then used when the program is run and LVLCHK(*RMTFILE) is in effect for the DDM file.

Whether or not DDS is used to describe a remote AS/400 file, a source system program can still have its own field and record format definitions provided in the program, or the program can substitute the definitions of another source system file that is created using DDS. Either can be done if LVLCHK(*NO) is in effect in the DDM file or specified in an Override with Database File (OVRDBF) command used at program run time. LVLCHK(*NO) need only be used when the record format used on the source system is different than that of the remote AS/400 file.

Non-AS/400 Target Considerations

DDS can be used with a non-AS/400 file only if the local AS/400 program is compiled using a local AS/400 file that has the same record format name as the DDM file being used. After the program is compiled, the local file can be overridden by a DDM file that accesses the remote file. LVLCHK(*NO) must be specified in the DDM file or in an OVRDBF command.

If no DDS exists on the local system to describe the remote file, the program must describe the fields. The Display File Field Description (DSPFFD) command can be used to display the field attributes of the remote file. LVLCHK(*NO) should be specified in the DDM file or in an OVRDBF command.

If LVLCHK(*RMTFILE) is specified or assumed, the program must be compiled (or recompiled) using a DDM file that accesses the remote file. The AS/400 system then creates a record format and fields for the remote file. The names of the fields that are created are of the type K*nnnnn* for keyed fields and F*nnnnn* for nonkeyed fields.

DDM-Related DDS Keywords and Information

All the information about DDS keywords that relates specifically to DDM is provided in this section.

- Considerations for creating local files:
  - The following DDS keywords *cannot* specify the name of a DDM file: REFACCPTH, and FORMAT.
  - The DDS keywords REF and REFFLD *can* specify the names of DDM files to refer to remote files; however, the remote files must be on an AS/400 system or a System/38. When a DDM file name is specified as the database file name in either keyword, it refers to the DDM file on the source system, and the referred to field name and record format name refer to a field and record format used in the remote file on the target system.
- Considerations for creating logical files when the remote system is not an AS/400 system:
  - At least one key field must be specified in the record format for the logical file.
  - Only one file can be specified on the PFILE keyword.
  - SELECT and OMIT functions are not supported.
  - Logical join files are not supported.
  - Field names of remote physical files have the naming convention of F00001, F00002, F00003, and so forth (F*nnnnn*) for nonkeyed fields and K00001, K00002, K00003, and so forth (K*nnnnn*) for keyed fields.

    The exception to this naming convention is when the target system is a System/38 and the physical file was created locally. In this case the field names are the same as the field names specified when the physical file was created.
  - All the fields defined for the logical file must be specified in the same order as defined in the physical file. This can be done by default.
  - The SST keyword can be used to access partial fields of the physical file. The use of two or more substring fields is

required to define the entire physical field. In addition, the partial fields must be in the same order as defined in the substring field of the physical file.

  - The CONCAT keyword can be used to group physical file fields into one logical field. The concatenation order of the fields must be in the same order as defined in the physical file.
  - The fields of the physical file must be specified in the same order as defined in the physical file.

- PFILE and JFILE considerations for creating remote files:
  - The record format name specified for the physical file in the DDM file on the JFILE or PFILE keyword must be the same name as the DDM file that represents the remote physical file.
  - When creating a logical file, the file specified on PFILE or JFILE must be a DDM file, and the location for each physical file in the DDM file on the JFILE or PFILE keyword must be the same as the location of the DDM file for the logical file. In other words, the physical files and logical file must be on the same remote system.

  If the remote system is a release 1.0 or 1.2 AS/400 system, attempting to create a file using the FCFO keyword will fail.

- When the system is not an AS/400 system, these keywords are either ignored or not supported for *logical* files:

ABSVAL	EDTCDE	LIFO
ACCPTH	EDTWRD	NOALTSEQ
ALIAS	FCFO	RANGE
ALL	FLTPCN	REFSHIFT
ALTSEQ	FORMAT	RENAME
CHECK	JDFTVAL	SIGNED
CMP	JDUPSEQ	TEXT
COLHDG	JFILE	TRNTBL
COMP	JFLD	VALUES
DIGIT	JOIN	ZONE
DYNSLT	JREF	

- When the system is not an AS/400 system, these keywords are either ignored or not supported for *physical* files:

ABSVAL	EDTCDE	RANGE
ALTSEQ	EDTWRD	RESHIFT
CHECK	FCFO	SIGNED
CMP	FLTPCN	TEXT
COLHDG	FORMAT	VALUES
COMP	LIFO	ZONE
DIGIT	NOALTSEQ	

DDM User Profile Authority

AS/400 system users are not allowed to perform functions equivalent to CL commands on remote AS/400 systems using DDM without the proper command authorization. The user profiles associated with the target jobs must have *OBJOPR authority to the following CL commands to start the equivalent operation on the remote AS/400 system:

Command Name	Descriptive Name
ADDLFM	Add Logical File Member
ADDPFM	Add Physical File Member
ALCOBJ	Allocate Object
CHGLF	Change Logical File
CHGLFM	Change Logical File Member
CHGPF	Change Physical File
CHGPFM	Change Physical File Member
CRTLF	Create Logical File
CRTPF	Create Physical File
DLTF	Delete File
INZPFM	Initialize Physical File Member
RGZPFM	Reorganize Physical File Member
RMVM	Remove Member
RNMM	Rename Member
RNMOBJ	Rename Object

Chapter 6. Operating Considerations for DDM

This chapter provides task-oriented information (with examples) that describes various aspects of DDM operation considerations.

This chapter tells how the AS/400 system functions, both as a source or target system, when it communicates with another AS/400 system to perform remote file processing. It also describes the significant differences when an AS/400 system is communicating with another system that is not an AS/400 system.

Note: Although this chapter contains information about systems other than the AS/400 system, it does not contain all the information that the other system types using DDM may need to communicate with an AS/400 system. For complete information about how DDM is used with a particular remote system, refer to that system's documentation.

Described in this chapter are:

- Remote file accessing considerations
- Remote file member accessing considerations
- File access methods used with DDM
- Other remote file functions
- Performance considerations
- Problem analysis on the remote system
- System/36 considerations
- Personal computer source considerations

Note: Before reading this chapter, you might want to read (or review) the information under "Additional OS/400 DDM Concepts" on page 1-10, particularly the information about DDM conversations and about source and target jobs.

File Access Considerations

The following sections describe the types of files supported by an AS/400 system, when the DDM file and remote file must exist, and how to specify the names of remote files. Also included are examples and considerations for AS/400-to-AS/400 and AS/400-to-System/36 file accessing.

Types of Files Supported by OS/400 DDM

OS/400 DDM supports all AS/400 file types when the target system is another AS/400 system. If the target system is not an AS/400 system, the corresponding file types may be known by different names on that system. The following table shows the AS/400 equivalents of non-AS/400 files and DDM architecture files:

AS/400 Types	Non-AS/400 and DDM Architecture Types
Nonkeyed physical file	Sequential (or direct) access file
Keyed physical file	Keyed access file
Logical file	Logical file

The following list describes the considerations that apply to the types of files supported by an AS/400 system.

- AS/400 multiple-format logical files are not supported by DDM when the source or target system is neither an AS/400 system nor a System/38.
- For target physical (sequential or direct) files, if a record number is specified that is past the end of the file, the file is not extended and an error occurs.
- For target nondirect sequential files, the Clear Physical File Member (CLRPFM) command does not prepare a file member with deleted records.
- DDM files can be used as data files or source files by high-level language (HLL) programs. However, when a DDM file is used as a source file, the target system must be an AS/400 system or a System/38 and the remote file associated with the DDM file must be defined on the target system as a source file. That is, the remote file must have been created on the target AS/400 system or the target System/38 as FILETYPE(*SRC) by the Create Physical File (CRTPF) command or with

FMTOPT(*CVTSRC) specified on the Copy File (CPYF) command.

For a list of control language (CL) commands that can support DDM files as source files, see "Source File Commands" on page 5-22.

Existence of DDM File and Remote File

A file on a target system cannot be accessed for any type of operation (such as open, read, write, or display) unless a DDM file associated with the remote file already exists on the source system. However, the remote file does not need to exist at the time that the DDM file is created or changed using the Create DDM File (CRTDDMF) command or the Change DDM File (CHGDDMF) command, because the remote file is not referred to until the DDM file is actually opened for access.

Specifying Target System File Names

The rules for specifying the name of a DDM file (on the local AS/400 system) are the same as for any other file type on an AS/400 system. The rules, however, for specifying the name of a remote file depend on the type of target system.

A remote file name can be specified only on the RMTFILE parameter of the Create DDM File (CRTDDMF) and Change DDM File (CHGDDMF) commands. Following are the maximum number of characters that can be used on the RMTFILE parameter to specify a remote file name:

- For the AS/400 system (database management): 33 characters. This maximum can occur when a full name is specified that includes a library qualifier and a member name. For example:

  ```
  LIBRARY123/FILE123456(MEMBER1234)
  ```

  The value DM can be added to the name to specify that this is a data management file. There can be one or more blanks between the name and DM. This is the default.

- For the AS/400 system (folder management services): 76 characters. This maximum can occur when a fully qualified path name (consisting of 76 characters) is specified. For example:

  ```
  /Path123/Path223/Path323/Path423/
     Path523/Path623/Path723/Path823/Path923/DOC1 FMS
  ```

  The value FMS specifies that this is a folder management object. There can be one or more blanks between the name and FMS.

- For System/38: 33 characters. This maximum can occur when a full name is specified that includes a library qualifier and a member name. For example:

  ```
  FILE123456.LIBRARY123(MEMBER1234)
  ```

- For System/36 and CICS: 8 characters. For example:

  ```
  FILE1234
  ```

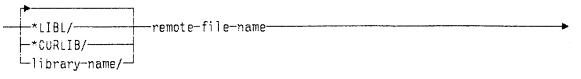
- For other systems: 255 characters is the maximum length allowed by the DDM architecture. The actual maximum length and syntax are determined by the target system.

Target AS/400 File Names: As with local files, every AS/400 remote file, library name, or member must begin with an alphabetic character (A through Z, $, #, or @) and can be followed by no more than 9 alphanumeric characters, A through Z, 0 through 9, $, #, @, _, or period (.). No name can exceed 10 characters. Blanks are not allowed in AS/400 names.

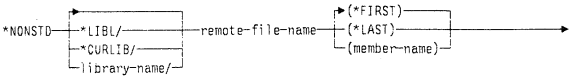
The use of an extended name allows additional graphic characters to be included in quotation marks ("). The extended name also cannot exceed 10 characters, but quotation marks are included with the name, thereby limiting the number of graphic characters to 8. Lowercase letters remain lowercase letters. Examples of extended names are as follows:

```
"Test.Job"
"()/+="
```

When an AS/400 system is the target system, the file name can be specified in various forms, as shown in the following syntax and examples.



or



library-name

Specifies the name of the library that contains the remote file. *LIBL causes the library list of the job on

the target system to be searched for the specified file name. *CURLIB specifies the current library on the remote system.

remote-file-name

Specifies the name of a database file (physical, logical, or source file) on the target AS/400 system.

***NONSTD** Specifies, for an AS/400 target, that a member name is being included with the name of the remote file. The value *NONSTD must precede the full name, and the full name must be enclosed in apostrophes and be in all uppercase.

Note: If you press F4 (Prompt) when on the Create DDM File or Change DDM File displays, and specify the *NONSTD value with the remote file name abcde, the system converts abcde to 'ABCDE' (all uppercase) and the request is processed. However, if there is a slash or parenthesis in the remote file name, the system puts apostrophes around the name but does not convert it to uppercase.

Therefore, if you are using the *NONSTD value for the remote file name and the target system requires uppercase file names, type the remote file name in uppercase characters even when using F4 (Prompt).

member-name

Specifies the name of the member in the remote file. The member name must be enclosed in parentheses and immediately follow the file name (with no space). If no member name is specified, then *FIRST is assumed and the first (or only) member in the file is accessed. This is the oldest (or only) member in the file.

*LAST is supported only on the Override with Database File (OVRDBF), Clear Physical File Member (CLRPFM), Initialize Physical File Member (INZPFM), Reorganize Physical File Member (RGZPFM), Open Database File (OPNDBF), and Open Query File (OPNQRYF) commands. *LAST is the newest (or only) member in the file.

The following are examples of valid AS/400 remote file names:

```
CUSTMAST
PRODLIB/CUSTMAST

*NONSTD 'CUSTMAST(MBR1)'
*NONSTD '*LIBL/CUSTMAST(MBR2)'
*NONSTD 'PRODLIB/CUSTMAST(MBR3) DM'
*NONSTD 'PRODLIB/CUSTMAST(*FIRST)'
```

Target Non-AS/400 File Names: For non-AS/400 remote file names, the name must be in the form required by the target system. If special characters are used in the remote file name, *NONSTD and apostrophes must be used to specify the name, as shown above for specifying an AS/400 member name. If the name string contains no more than 10 characters and no special characters, it can be entered without the *NONSTD value and the apostrophes.

Using Location-Specific File Names for Commonly Named Files: When multiple systems are involved in a network, naming DDM files with location-specific file names can reduce confusion about which target system is being accessed for a file that has a commonly used name. For example, for an inventory file that may be named INVEN on multiple systems, using location-specific names such as NYCINVEN, STLINVEN, and DALINVEN for the DDM files on the local system to access files in New York City, St. Louis, and Dallas helps you to access the correct file.

Using an abbreviation or code that identifies the destination target system as part of the DDM file names makes it easier to remember where the desired remote file is located.

For non-AS/400 remote files that have record formats, using the same name for the DDM file as for the record format can also be useful.

Examples of Accessing AS/400 Remote Files (AS/400-to-AS/400)

The following examples show how access to a DDM file becomes an indirect reference (via DDM) to the actual file on some other system. These examples are AS/400-to-AS/400 examples.

Note: All of these examples assume that the DDM file on the local AS/400 system is named DDMLIB/RMTCAR and that it is associated with a remote file named SALES/CAR on an AS/400 system in Chicago.

- **Creating a DDM file to access a remote file:**

```
CRTDDMF   FILE(DDMLIB/RMTCAR)  RMTFILE(SALES/CAR)
   RMTLOCNAME(CHICAGO)  TEXT('Chicago file SALES/CAR')
```

  This command creates a DDM file named RMTCAR and stores it in the DDMLIB library on the local system. The remote file to be accessed is the CAR database file in the SALES library on the Chicago system. (The remote file is *not* accessed at the time the Create DDM File [CRTDDMF] command is used to create the DDM file. The existence of the file SALES/CAR is not checked when the DDM file is created.) Later, when the DDM file is accessed by a local program, the remote location CHICAGO is used by DDM to access the SALES/CAR file on the Chicago system.

- **Copying a local file to a remote file:**

```
CPYF   FROMFILE(QGPL/AUTO)  TOFILE(DDMLIB/RMTCAR)
```

  This command copies the data from the AUTO file in the QGPL library on the local system into a remote file named SALES/CAR on the Chicago system, via the DDM file DDMLIB/RMTCAR.

- **Allocating a remote file and member for use:**

```
ALCOBJ   OBJ((DDMLIB/RMTCAR  *FILE  *EXCL))
```

  The Allocate Object (ALCOBJ) command is used to allocate (lock) both the DDM file (RMTCAR) on the source system and the first member of the remote file (as well as the file itself) on the target system. In effect, the command

```
ALCOBJ   OBJ((SALES/CAR  *FILE  *EXCL  *FIRST))
```

  is run on the target system.

- **Overriding a local file with a DDM file:**

```
OVRDBF   FILE(FILEA)  TOFILE(DDMLIB/RMTCAR)
   POSITION(*RRN 3000)
```

  This command overrides the database file FILEA with the DDM file RMTCAR, stored in the DDMLIB library. Both files are on the source system. Whatever remote file is identified in the DDM file (in this case, SALES/CAR on the Chicago system) is the file actually used by the source system program. When the remote file is opened, the first record to be accessed is record 3000.

- **Displaying records in a remote file:**

```
DSPPFM   FILE(DDMLIB/RMTCAR)
```

  This command displays the records in the first member of the remote file SALES/CAR, which is associated with the DDM file DDMLIB/RMTCAR.

- **Displaying the object description of a DDM file:**

```
DSPOBJD   OBJ(DDMLIB/RMTCAR)  OBJTYPE(*FILE)
```

  This command displays, on the local system, the object description of the RMTCAR DDM file. No reference is made by this command to the associated remote file on the Chicago system.

- **Displaying the file description of a DDM file:**

```
DSPFD  FILE(DDMLIB/RMTCAR) TYPE(*ATR) FILEATR(*DDM)
   SYSTEM(*LCL)
```

  This command displays, on the source system, the file description of the DDM file named RMTCAR in the DDMLIB library. As indicated by the TYPE parameter, the attributes of the DDM file are displayed. Only the DDM file's attributes are displayed because FILEATR(*DDM) is specified.

  Because SYSTEM(*LCL) is specified, the attributes of the DDM file are displayed and the remote system is not accessed. If SYSTEM(*RMT) is specified, the attributes of the associated remote file are displayed. If *RMT or *ALL is specified, the remote system is accessed to get the attributes of the remote file.

- **Deleting a DDM file:**

```
DLTF  FILE(DDMLIB/RMTCAR)  SYSTEM(*LCL)
```

  This command deletes the DDM file on the local system. Again, no reference is made to the associated SALES/CAR file on the Chicago system. If SYSTEM(*RMT) or SYSTEM(*FILETYPE) is specified, SALES/CAR on the Chicago system would be deleted.

Example of Accessing System/36 Remote Files (AS/400-to-System/36)

Of the command examples given in the previous topic (showing AS/400-to-AS/400 examples), all except the first example can be coded the same way for accessing a file on a System/36. That is, if the remote file name SALES/CAR is changed to CAR to meet the System/36 naming conventions, all the commands (except the first) can be used without change to access a remote System/36 file instead of an AS/400 file.

The first example from the previous section is recoded here to access a remote System/36 file. Besides changing the remote file name, another parameter that should be coded is LVLCHK(*NO).

```
CRTDDMF   FILE(DDMLIB/RMTCAR)  RMTFILE(*NONSTD 'CAR')
   RMTLOCNAME(CHICAGO)  TEXT('Chicago file CAR on S/36')
   LVLCHK(*NO)
```

This command creates a DDM reference file named RMTCAR and stores it in the DDMLIB library on the local AS/400 system. The remote file to be accessed is the CAR file on the System/36 named CHICAGO. LVLCHK(*NO) is specified to prevent level checking because the level identifiers created for the System/36 file do not match those in the program when it accesses the file.

Member Access Considerations

Members are supported for database I/O operations only if the target system is an AS/400 system or a System/38. Members are not supported if the target system is neither an AS/400 system nor a System/38.

Members can be locked before use, using the Allocate Object (ALCOBJ) command if the target system is an AS/400 system or a System/38.

The DDM file itself does not have members like a database file. However, if a member is identified on the source system (for example, using the Override with Database File [OVRDBF] command) and the target system is an AS/400 system or a System/38, that member name is used to identify a member in the target system's file. When the target system is neither an AS/400 system nor a System/38, and if the member name is specified as *FIRST, or in some cases *LAST, or the file name is the same as the member name, then the RMTFILE parameter values in the DDM file are sent without change. This allows file access on systems that do not support members.

If the member name is other than *FIRST or in some cases *LAST, or the file name is different from the member name (for example, when the file is opened) and the target system does not support members, an error message is sent to the requesting program and the function is not performed.

Examples of Accessing Remote Members (AS/400 System Only)

The following examples show how access to a DDM file becomes an indirect reference (via DDM) to a member of a file on a remote AS/400 system. These examples are AS/400 system-to-AS/400 system examples.

```
CRTDDMF   FILE(DDMLIB/RMTCAR)  RMTFILE(SALES/CAR)
   RMTLOCNAME(CHICAGO)
OVRDBF    FILE(FILE1)  TOFILE(DDMLIB/RMTCAR)  MBR(TEST1)
OVRDBF    FILE(FILE2)  TOFILE(DDMLIB/RMTCAR)
```

This example shows the creation of the same DDM file as in the previous examples. Then, OVRDBF commands are used to override two local files named FILE1 and FILE2 with the local DDM file RMTCAR. When an application program attempts to open the files, the DDM file DDMLIB/RMTCAR is opened twice instead. (FILE1 and FILE2 are not opened.)

Once communications are established with the correct target system, the target system's TDDM opens the remote file SALES/CAR twice (two recursions) and opens two different (in this case) members in that file: member TEST1 and member *FIRST (the first member). This example requires only one DDM conversation and one target job because both open operations use the same DDM file and, therefore, the same location.

```
CLRPFM    FILE(DDMLIB/RMTCAR)  MBR(FRED)
```

This command clears, via the DDM file named DDMLIB/RMTCAR, member FRED of the file SALES/CAR on the target system.

Example of a DDM File That Opens a Specific Member

A specific file member can be specified in the RMTFILE parameter, which is used on only the Create DDM File (CRTDDMF) and Change DDM File (CHGDDMF) commands, by using the *NONSTD value followed by the file, library, and member name. This allows an application program to process a member other than the first member (*FIRST) without using file overrides. However, if the program requires redirection to more than one member, overrides should be used. Also, programs that already use overrides to specify members of local files should continue to do so, even if overrides to remote files are also used; otherwise, programs that worked locally would no longer do so. If the RMTFILE parameter contains a member name and an override with a different member name is in effect, the file open requests fails.

Note: If you press F4 (Prompt) when on the Create DDM File or Change DDM File displays, and specify the *NONSTD value with the remote file name abcde, the system converts abcde to 'ABCDE' (all uppercase) and the request is processed. However, if there is a slash or parenthesis in the remote file name, the system puts single quotation marks around the name but does not convert it to uppercase.

Therefore, if you are using the *NONSTD value for the remote file name and the target system requires uppercase file names, type the remote file name in uppercase characters even when using F4 (Prompt).

```
CRTDDMF   FILE(DDMLIB/RMTCAR)  RMTFILE(*NONSTD
   'SALES/CAR(JULY)')  RMTLOCNAME(CHICAGO)
```

When a program opens the DDM file named RMTCAR on the source system DDMLIB library, the target AS/400 system opens the member JULY in the file SALES/CAR.

Access Method Considerations

Basically, access methods control what subsets of functions can be performed after a particular remote file is opened. This may mean that an AS/400 program, or group of programs sharing a non-AS/400 file, cannot do all the same operations that are possible using a file that is on the local AS/400 system.

For example, assume that an AS/400 application program opens a keyed file with SHARE(*YES) and performs keyed I/O operations. It then calls another program that does relative record number operations using the same open data path (ODP) (because SHARE was specified). **Relative record numbers** specify the relationship between the location of a record and the beginning of a database file, member, or subfile. If the first program is redirected by an Override with Database File (OVRDBF) command to use a remote keyed file on a System/36, this scheme no longer works. If a *keyed* access method is selected, record number operations fail. If a *record number* access method is selected, keyed operations fail.

Notice that when both source and target systems are AS/400 systems, access methods are not used. A potential problem exists when the target system is neither an AS/400 system nor a System/38. Notice also that the combined-access access method (*COMBINED) is not supported by System/36, and probably not by any target other than an AS/400 system or System/38.

Access Intents

When a program opens a file, it must specify how it intends to work with the records in the file: read, add, update, delete, or a combination of these. Of course, to successfully perform these operations, the job and/or user running the program must have the corresponding data authorities. The AS/400 system does not check to make sure all data authorities exist when the file is opened, but it does check for each required data authority when the corresponding I/O operation is done using the file. The System/36 does check these data authorities at open time; therefore, a program may no longer work using a remote file on a System/36, even though the requester's data authorities to the

remote file are the same as for a local file (which will work).

For example, assume that a program is used by two groups of users on an AS/400 system to access the same local AS/400 file. Group A has only *READ authority, while group B has *READ, *ADD, and *UPDATE. The program always opens the file for *READ, *ADD, and *UPDATE. But it has a *read only* logic path that is used when a member of group A calls the program. In this way, no authority exceptions are encountered, even though exceptions would be created if members of group A attempted to add or update records. Now, if this program is redirected to a remote System/36 file to which members of both user groups have the same data authorities as they had to the local AS/400 file, the program may not work for members of group A. This is because the System/36 may reject requests to open the file when the requester does not have data authorities matching those specified in the access intent list accompanying the open request.

Key Field Updates

An AS/400 program is allowed to change any part of a data record including key fields. The exception to this is a COBOL/400 program because the COBOL/400 language does not allow key field changes. A System/36 program cannot change primary key fields in a record, regardless of the access method specified when the file is opened. Logical file key fields can be changed under some circumstances, but primary key fields can never be changed.

This means that an RPG/400 program, for example, that routinely changes key fields in a local keyed file may fail when it is redirected to a remote keyed file on a System/36 (or other system with similar restrictions). Several different errors may be returned by the DDM target, depending on the access method or access path being used when the key field change is attempted.

Deleted Records

On the AS/400 system, a record is marked as deleted by the system. This is done either when an active record is deleted by an application or when a file is created with deleted records (for example, with the Initialize Physical File Member [INZPFM] command). A record that is added to a file or changed in a file is never marked as deleted, unless a subsequent delete operation is performed. On some other systems, like the System/36, a special data value in the record may be used to indicate deleted status. For example, if a record contains all hex FFs, it may be considered deleted.

This means that an AS/400 application normally used to add or change records in a local file may encounter errors when attempting these operations with a remote file on a system that is neither an AS/400 system nor a System/38. If the application happens to supply a record that is considered deleted by the target DDM system, the target may reject the add-or-change request.

Blocked Record Processing

If SEQONLY is used to block records sent to a remote system, the records are not sent until the block is full. If a source job is canceled before a block is sent, the records in the block are lost. If blocking is used, the user should make sure a force end of data or close of the file is done before canceling the source job.

Other DDM-Related Functions Involving Remote Files

Besides accessing remote files for data record I/O operations, other operations related to remote files can be performed. These are briefly described in the following sections.

Performing File Management Functions on Remote Systems

OS/400 DDM supports creating, deleting, or renaming of files on a remote system. The Submit Remote Command (SBMRMTCMD) command can be used to submit these types of file management commands, or other CL commands, to the target system so they can run on that system. The Submit Network Job (SBMNETJOB) command or display station pass-through can also be used, without the need for DDM.

Examples of how the SBMRMTCMD command can be used are provided in the topic "SBMRMTCMD (Submit Remote Command)

Command" on page 5-2 and in the task examples in Appendix A.

For all the functions that can be performed with the SBMRMTCMD command, refer to the CL command lists under "Object-Oriented Commands" on page 5-20, or refer to the summary chart in Appendix B.

Note: The CL commands in "Target AS/400-Required File Management Commands" on page 5-21, "Member-Related Commands" on page 5-21, and "Source File Commands" on page 5-22 do not need to be used with the SBMRMTCMD command; they can run directly on the target system by specifying a DDM file name on the CL command itself.

Locking Files and Members for DDM

Putting object locks on DDM files and their associated remote files requires special consideration.

Allocate Object (ALCOBJ) and Deallocate Object (DLCOBJ) Commands: The ALCOBJ command locks DDM files on the source system and the associated remote files on the target systems. When the target is an AS/400 system or a System/38, resulting locks on the remote files are the same as if the files were local files. When the target is neither an AS/400 system nor a System/38, equivalent locks are obtained, although the target system may promote the lock to a stronger lock condition than was specified on the ALCOBJ command.

Note: On systems that are neither AS/400 nor System/38 target systems, remote *files* are locked with the specified lock condition, and on AS/400 and System/38 target systems only, remote *members* are locked with a minimum specified lock condition. (AS/400 or System/38 remote *files* are locked with shared-read locks.)

For more information on these commands, see the topics "ALCOBJ (Allocate Object) Command" on page 5-9 and "DLCOBJ (Deallocate Object) Command" on page 5-14.

Work with Job (WRKJOB) and Work with Object Lock (WRKOBJLCK) Commands: For both the WRKOBJLCK command and menu option 12 (Work with locks, if active) of the WRKJOB command, only the locks held for the local DDM files are shown, *not* locks held for the remote files (or for their members). If locked, DDM files are always locked as shared read (*SHRRD), regardless of the lock conditions used for their associated remote files or members.

For more information on these commands, see the topics "WRKJOB (Work with Job) Command" on page 5-17 and "WRKOBJLCK (Work with Object Lock) Command" on page 5-17.

Controlling DDM Conversations

Normally, the DDM conversation(s) associated with a source system job is kept active until:

1. All the DDM files and remote files used in the conversation are closed and unlocked (deallocated).
2. No other DDM-related functions like the use of the Submit Remote Command (SBMRMTCMD) command or the Display File Description (DSPFD) command (to display remote file information) are being performed.
3. No DDM-related function has been interrupted (by a break program, for example) while running.
4. The ENCMTCTL command was issued (if commitment control was used with a DDM file).
5. No distributed relational database architecture-related functions are being performed.
6. The job or routing step ends.

If *1*, *2*, and *3* are true and the source job has not ended, the conversation is considered to be *unused*; that is, the conversation is kept active but no requests are being processed.

DDM conversations can be active and unused because the default value of the DDMCNV job attribute is *KEEP. This is desirable for the usual situation of a source system program accessing a remote file for multiple I/O operations; these operations are handled one at a time, as shown in Figure 1-7 on page 1-12 and explained in the text following it.

If multiple DDM requests are to be made in a job and the DDM files are being continually opened and closed in the job, *KEEP should be used to keep an unused DDM conversation active.

(However, as long as one DDM file remains open or locked, *KEEP has no effect.)

For source jobs that access remote files but do not access data records in them, it may be desirable, depending on the frequency of the file accesses, to automatically drop each DDM conversation at the completion of each file-related source job request. Whether the conversation in the source job is kept active or automatically dropped during the time a conversation is unused is determined by the DDMCNV job attribute value (*KEEP or *DROP). See "DDMCNV Parameter Considerations" on page 5-18 for the description of these values.

Regardless of the value of the DDMCNV job attribute, conversations are dropped at the end of a job routing step, at the end of the job, or when the job initiates a Reroute Job (RRTJOB) command. Unused conversations within an active job can also be dropped by the Reclaim DDM Conversations (RCLDDMCNV) or Reclaim Resources (RCLRSC) command. Errors, such as communications line failures, can also cause conversations to drop.

Displaying DDMCNV Values (WRKJOB Command): To display the current value (*KEEP or *DROP) of the DDMCNV job attribute for your source job, you can use menu option 2 (Work with definition attributes) on the Work with Job (WRKJOB) Command display. You can also find out the value within a CL program by using the Retrieve Job Attributes (RTVJOBA) command.

Changing DDMCNV Values (CHGJOB Command): To control whether the system is to automatically reclaim (or drop) DDM conversations in a source job whenever they become unused, the system default *KEEP can be changed to *DROP by using a Change Job (CHGJOB) command. If the value is left as *KEEP, the Reclaim DDM Conversations (RCLDDMCNV) or Reclaim Resources (RCLRSC) command can be used at any time to drop all DDM conversations (within that job only) that currently do not have any active users.

Reclaiming DDM Resources (RCLRSC and RCLDDMCNV Commands): When an AS/400 user wants to ensure that the resources for all APPC conversations (including DDM conversations) that are no longer active are returned to the system, the Reclaim Resources (RCLRSC) command can be used. To reclaim currently unused DDM conversations in a job, use the Reclaim DDM Conversations (RCLDDMCNV) command. The DDM-related information about these commands is described in Chapter 5. For complete non-DDM-related information about these commands, refer to the *CL Reference* manual.

Displaying DDM Remote File Information

The CL commands Display File Description (DSPFD) and Display File Field Description (DSPFFD) can be used by an AS/400 source system user to display the attributes of one or more DDM files on the source system, or to display the attributes of one or more remote files on a target system. See the topics "DSPFD (Display File Description) Command" on page 5-15 and "DSPFFD (Display File Field Description) Command" on page 5-15 for how this is done.

Displaying Remote File Records

The Display Physical File Member (DSPPFM) command can be used to display a remote file on a target system. For performance reasons, however, whenever possible, you should use display station pass-through to sign on the remote system, and display the file directly. When display station pass-through is used, only the display images are transmitted over the communications line. When DDM is used to access the remote file, each record is transmitted separately over the line, which requires many more transmissions.

If pass-through cannot be used (for example, if the remote file is not on an AS/400 system, a System/38, or a System/36, or if pass-through is not configured on your system), direct record positioning rather than relative positioning should be used whenever possible. For example, if record number 100 is being displayed and you want to see record number 200 next, that record is accessed faster if you enter

200 in the control field instead of +100. The results are the same, unless the file contains deleted records.

Using Object Distribution

Although DDM file names can be specified on the Send Network File (SNDNETF) and Receive Network File (RCVNETF) commands, these commands should be run, whenever possible, on the system where the data actually exists. Therefore, if both systems are AS/400 systems and both are part of a SNADS network, **object distribution** can be used instead of DDM to transfer the data between them.

- The SNDNETF command should run directly on the system that contains the data being sent. If necessary, the Submit Remote Command (SBMRMTCMD) or Submit Network Job (SBMNETJOB) command can be used to submit the SNDNETF command to the system where the data exists.

  Note: Another way to use the SNDNETF command without using DDM is to run it on the target system using display station pass-through.
- The RCVNETF command must be run on the system where the data has been sent. If necessary, a DDM file may be referred to on the RCVNETF command to place the data on another system. However, if possible, you should arrange to have the data sent to the system where the data is to be used, to avoid using a DDM file.

For both sending and receiving operations, the file types of the data files must match and can only be a save file or a physical database file. If DDM is being used, however, the file being transferred cannot be a save file.

Using Object Distribution with DDM

You can also use *both* SNADS (on AS/400 systems) and DDM (on AS/400 systems and non-AS/400 systems) to transfer files between AS/400 systems and systems that are not part of a SNADS network but that do have DDM installed. (Although a System/36 may have SNADS, it cannot be used for AS/400 object distribution.)

For example, if an OS/400 DDM file refers to a file on a System/36, the AS/400 system can use the SNDNETF command to send the file to another AS/400 system using object distribution. Similarly, if a file has been sent to an AS/400 system, the RCVNETF command can be used to receive the file onto a System/36 using DDM.

For more information on using object distribution with SNADS, see the *Distribution Services Network Guide*.

Performance Considerations

This section provides information to help you improve performance when using DDM and also provides some information about when to use something other than DDM to accomplish some functions.

- When a DDM file is specified on the CPYF command, optimal performance is obtained if the following are all true:
  - The from-file is a logical or physical file and the to-file is a physical file.
  - FMTOPT is *NONE, *NOCHK, or not specified.
  - INCHAR, INCREL, ERRLVL, RCDDMT (*ALL), PRINT(*COPIED), PRINT(*EXCLD), SRCSEQ, TOKEY, SRCOPT, or FROMKEY parameters are not specified.
  - The from-file is not overridden with the POS keyword, other than *NONE or *START.
  - The to-file is not overridden with INHWRT(*YES).
- The Create Query Application (CRTQRYAPP) or Open Query File (OPNQRYF) command uses System/38 extensions to the DDM architecture. When either the source or the target system is neither an AS/400 system nor a System/38, these System/38 extensions to the DDM architecture are not used. The System/38 DDM architecture extensions minimize DDM system processing time.
- In the case of query functions (AS/400 command CRTQRYAPP OPTIMIZE[*YES]), this significantly reduces the amount of data transferred between the systems. However, for user-written applications, the amount of data exchanged between the systems is larger than that used to communicate using DDM with non-AS/400 systems. The addi-

tional data provides AS/400 extended DDM functions and also reduces source system DDM processing overhead. Using normal read, write, update, add, and delete operations as examples, consider the following:

  - Standard DDM architecture DDM overhead data includes such information as a file identification, operation code, and simple result information. A user program read-by-key operation uses approximately 40 characters of DDM information in addition to the length of the key data. Data returned from the remote system uses approximately 32 characters of DDM information plus the length of the data file record.
  - System/38 DDM architecture extensions cause additional data overhead such as record format identification and a major portion of the I/O feedback area information. A user program read-by-key operation uses approximately 60 characters of DDM information in addition to the length of the key data. Data returned from the remote system uses approximately 80 characters of DDM information plus the length of the data file record. Normally the additional length in data streams is not noticeable. However, as line activity increases, line utilization may peak sooner when using these extended data streams versus standard DDM data streams.

- The target DDM job priority is controlled by the job class specified by the associated subsystem description routing entry. The following routing entry is normally the one used for all target (program start request) jobs:

  ```
  ADDRTGE ... PGM(*RTGDTA) ... CMPVAL(PGMEVOKE 29)
  ```

  The subsystems QBASE and QCMN, which are shipped with the AS/400 system, have this routing entry.

  To have target DDM jobs in a subsystem run at a different priority than other APPC target jobs in the same subsystem, insert the following routing entry with the appropriate sequence number:

  ```
  ADDRTGE SBSD(xxx) SEQNBR(nnn) CMPVAL(QCNTEDDM 37)
          PGM(*RTGDTA) CLS(uuu)
  ```

  The class *uuu* is used to determine target job priority.

- Using the get and get graphic functions of the OfficeVision/400 word processing function to retrieve large amounts of data may cause serious performance effects. For more information, see "OfficeVision/400" on page 2-14.
- To display records in a remote file, display station pass-through should be used whenever possible. Otherwise, direct record positioning should be used with the Display Physical File Member (DSPPFM) command, as described under "Displaying Remote File Records" on page 6-9.
- If a DDM user exit security program (described in Chapter 4) is a CL program and it creates an OS/400 exception and an inquiry message that requires the target system operator to respond, both the user exit program and the source system job must wait for the response. Consider using the default system reply list by specifying INQMSGRPY(*SYSRPYL) for the TDDM job's description specified on the Add Communications Entry (ADDCMNE) command for that APPC remote location information. See "User Exit Program Considerations" on page 4-8 for more information.
- The WAIT and WAITFILE parameters, used on commands like Allocate Object (ALCOBJ) or Receive Message (RCVMSG), have no effect on the source system program. These parameters function the same as they do when local files are accessed. The wait time values specified on commands run on the source system do not take effect on the source system; they affect only the target system and only if it is an AS/400 system or a System/38.

  Notes:

  1. The WAITFILE parameter of the create or change OS/400-Intersystems Communications Function (ICF) file command determines how long the APPC support will wait for session resources to become available when doing an acquire operation or a start function. The WAITFILE value is not used for sessions where the connection to the adjacent system is over a switched connection; for example, an SDLC switched line, an X.25 SVC line, an Ethernet line, or a token-ring connection. Using a switched con-

nection, acquire operations and start functions do not time out.

2. Because APPN sessions may cross multiple systems and lines to reach the remote system, the WAITFILE timer should be adjusted to allow more time in these cases. You should not specify *IMMED for the WAITFILE parameter if your application is running in a network configured to use APPN functions. The value you specify for this parameter is dependent on the size and type of the network.

- As for any LU session type 6.2 data exchange, certain SNA parameters can affect performance. These parameters include the path information unit size (MAXFRAME), the request/response unit size (MAXLENRU), SNA pacing (INPACING, OUTPACING), and, for X.25, packet size and window size. In general, the larger the value used, the better the realized performance.

- **SNA path information unit size**

  The path information unit (PIU) is the size of the actual data transmission block between two systems. The MAXFRAME parameter on the Create Controller Description (APPC) (CRTCTLAPPC) or Create Controller Description (SNA Host) (CRTCTLHOST) command specifies the path information unit size the local system attempts to use. During session establishment, both systems determine which size is used, and it is always the smaller value. See the *Communications Management Guide* for additional considerations on PIU size. Other remote systems may have different PIU size considerations.

- **SNA response/request unit size**

  The response/request unit (RU) size (CRTMODD MAXLENRU) controls the amount of system buffering before fitting that data into the path information unit that is actually transmitted. In APPC the transmit and receive RU lengths are negotiated during session establishment. Again, the negotiation results in the smallest value being used. See the *APPC Programmer's Guide* for additional considerations on RU size. Other remote systems have different RU size considerations.

- **SNA pacing values**

  The pacing value determines how many request/response units (RUs) can be received or sent before a response is required indicating buffer storage is available for more transmissions. During session establishment, both systems determine which size is used, and it is always the smaller value.

  In cases where both batch and interactive processing occur at the same time on the same communications line, AS/400 job priority may be used to favor interactive processing over batch processing. In addition, reducing the value of pacing for a batch application and raising it for an interactive application may be necessary to provide a level of line activity priority for the interactive application.

  On an AS/400 system, different pacing values can be specified through the creation of different MODES (Create Mode Description [CRTMODD] command) to the different applications. See the *APPC Programmer's Guide* for additional considerations on pacing values. Other remote systems have different SNA pacing value considerations.

- **X.25 packet**

  An X.25 packet smaller than the MAXFRAME value adds data transmission time over a non-X.25 data link. In general, for X.25, the longer the MAXFRAME and the actual amount of data being transmitted, the greater this difference is. In the case of DDM, which adds DDM control information to the normal file record data, the packet size has an additional effect on the difference between local and remote file processing and between non-X.25 and X.25 data links.

  In cases of many deblocked DDM operations, the number of packets required to transmit data may become so high that packet processing overhead within the X.25 adapter affects performance significantly. Use the largest X.25 packet window size supported by the network and communicating products to maximize performance.

  When X.25 must be used to access remote files, successive transmission of many small deblocked records, such as less than 80 character records, may cause the X.25 adapter to expend a disproportionate amount of time processing X.25 packet characters

versus transmission of user data. See the *X.25 Network Guide* for additional X.25 considerations. Other remote systems may have different packet window size considerations.

In general, the overhead in processing X.25 packets results in less throughput than the use of a conventional line when identical line speeds are used and data transfer is in only one direction. When data is transferred at the same time in both directions, the advantages of X.25 duplex support is realized. On the System/38, the overall processing effect is minimal, because the overhead in processing the packets is done within the Integrated X.25 Adapter.

In general, the processing of remote files via DDM is transparent to an application program or utility function, such as that provided by the Copy File (CPYF) command. However, additional time is required when accessing remote files via a communications line. The performance difference between local file and remote file processing is proportional to the number of accesses to remote files, the data record length, and the line speed during a unit of performance measurement.

An additional difference between local and remote file processing is that the input or output operation to a local file may not result in an immediate physical disk operation because the system transfers blocks of data from the disk and writes blocks of data to the disk. There are times, then, that the user program accesses data within main storage and the physical I/O occurs at a different time. Therefore, to minimize the difference between local file and remote file performance, it is essential that knowledge of an application design and the amount and type of accesses to files be considered when determining which files are to be accessed remotely using DDM.

The additional time for each remote access is comprised of:

- Additional system processing to convert local system file interfaces to the DDM architecture interfaces
- Amount of data transmitted over the communications line
- Amount of remote system processing of the file operations
- Speed of the communications line

The communications line time accounts for most of the additional time, though the actual time is dependent on line speed and the amount of line activity during the DDM function.

As is true in non-DDM cases, local and remote system job priorities have the most significant effect on performance. On an AS/400 system, the PRIORITY and TIME SLICE values of the class being used control job priority. The SDDM runs under the source job, and the TDDM runs under the class assigned to the APPC routing entry of the target system's subsystem. In applications that access multiple files, the best results are achieved when the most heavily accessed files are on the same system as the program that is running and the less heavily accessed files are on a remote system. Key considerations regarding the placement of files and application programs follow:

- The system having primary responsibility for file maintenance needs to be identified. In all cases of multiple systems applications, the best performance results if only one system is responsible for file maintenance. If an application program maintains the file through exclusive (nonshared) processing, best performance can be realized when the application program resides on the system with the file.

  In some cases, transmitting the file back to the local system may require:

  - An APPC program.
  - A program using remote DDM files.
  - The Copy File (CPYF) command via DDM.
  - Object distribution SNDNETF and RCVNETF operations. In interactive applications, display station pass-through should be considered when the amount of display data transferred is significantly less than the amount of database file data that would be sent via DDM.
- In cases where file placement requires movement of application processing to a remote system for best performance results, use of the Submit Remote Command (SBMRMTCMD) command should be considered. This works best in a batch processing input stream where each program waits for the preceding program to complete. The use

of the SBMRMTCMD command is valid only when the source and target systems are AS/400 systems or Systems/38s. For example, assume that program A accesses local files. Program A would run on a local system. Program B accesses remote files. You can use the SBMRMTCMD command to run program B on the remote system.

- In cases where file maintenance is shared across systems, the best performance can be obtained if the file is placed on the system with the largest percentage of file update, add, and delete operations.

  In certain cases, a pair of source and target APPC programs can provide performance improvements over DDM. For example, assume 10 records are to be retrieved from the remote system. When DDM is used and record blocking cannot be used (for example, user program random input operation, sequential input for change, or use of the OVRDBF SEQONLY[*NO] command), assume 10 data transmissions are sent and 10 are received, for a total of 20 transmissions. User-written APPC programs can build additional intelligence into the data stream such that request for the data and receipt of the data can be done in two data transmissions instead of 20, one request for all records of customer 00010 and one response containing 10 records for customer 00010. For additional information on the effects of record blocking on DDM, see the *Database Guide*.

Consider two sample application processing techniques, one batch file processing and the other interactive processing.

Batch File Processing: Consider the following when using batch file processing with DDM:

- When an application opens a local file for *sequential input only* or *output add*, the system uses blocking techniques to achieve maximum throughput. To ensure blocking is used for a remote file accessed via DDM, do not use random record processing operations in the program but specify OVRDBF SEQONLY(*YES) for the remote files opened by the program.
- Use of read and read-next operations in the high-level language (HLL) program to access the file maximizes the effect of the SEQONLY(*YES) specification.
- The use of random processing operations, such as chain operations of RPG/400 start operations of COBOL/400 programming language, causes DDM to send deblocked operations across the communications line even if the application processes the file data sequentially. This results in significant differences between local and remote file processing.
- When simple physical file transfer is desired (all records transferred and no application processing of the data), use of DDM via the Copy File (CPYF) command, or a user-written program using DDM with the Override Database File (OVRDBF) command SEQONLY(*YES number-of-records) specified, transfers the data more quickly than a user-written APPC program. The Copy File command and the DDM SEQONLY(*YES) support require less calls and returns between the program and APPC data management modules than does a standard RPG/400 or COBOL/400 APPC program.
- For RPG/400 or COBOL/400 sequential input-only applications, SEQONLY(*YES) should be specified with no *number of records* to achieve best throughput. For RPG/400 or COBOL/400 sequential output-only applications to keyed files, a large *number-of-records* value should be used. Refer also to the *Communications Management Guide* for considerations when using the SEQONLY parameter of the Override Database File OVRDBF command.
- The Send Network File (SNDNETF) command can be considered as an alternative to DDM or user-written APPC programs when transferring all records within a file to a remote AS/400 system. The SNDNETF command requires SNADS to be configured on the source and target AS/400 system. If one or more intermediate systems are between the source and target AS/400 systems, SNADS provides intermediate node routing of the data when correctly configured.
- Use of the SNDNETF command via SNADS offers the advantages of transmitting one copy of the data to multiple users on one or more target systems through a multiple node network, and the time scheduled trans-

mission of that data via the SNADS distribution queue parameter.

However, in addition to requiring SNADS to use the SNDNETF command, the target system user must also run the Receive Network File (RCVNETF) command to make the file usable on the target system. Use of DDM would not require this additional target system processing. For further information on Object Distribution and SNADS, refer to the *Distribution Services Network Guide* or see the topic "Using Object Distribution" on page 6-10.

In general, the file transmission times via SNADS (user program DDM sequential file processing, the DDM Copy File command, and a user-written APPC program between two AS/400 systems) are within 10% of each other. However, the use of the SNDNETF and RCVNETF commands to make a copy of the data usable on the target system does add total processing time over the other methods of file transfer.

- Because the SNDNETF command can transmit objects within a save file, the amount of data that is actually sent via this technique may be less than that sent using the other techniques. If the database file data sent contains a significant number of duplicate character strings, use of the Save Object (SAVOBJ) command parameter DTACPR(*YES) (data compression) can significantly reduce the amount of data that is actually sent via a SNADS distribution. However, if there are few duplicate character strings, there is little change in the amount of data sent.
- The AS/400 file transfer subroutines may also be used to transfer an entire file between AS/400 systems and an AS/400 system and a System/36. These subroutines may be called from high-level language programs, and in some cases throughput is achieved similar to that via DDM. See the *ICF Programmer's Guide*.

Interactive File Processing: Consider the following when using interactive file processing with DDM:

- The greater the number of random file operations per unit of performance measurement, the greater the difference between local and remote file processing because each operation has to be sent separately across the communications line. DDM cannot anticipate the next operation.

  Using a simple inquiry application that produces display output, via work station subfile support (as an example), consider an application that does 2 random record retrievals per Enter key versus one that does 15 random record retrievals. The operator may barely notice a delay in response time when 2 records are retrieved. However, there would be a noticeable difference between local and remote response time when 15 records are retrieved randomly from the remote system.
- Use of display station pass-through should be considered when the amount of data transferred back to the local (source) system per unit of performance measurement significantly exceeds the amount of data presented on the display. Test results have shown that the total elapsed time between a single deblocked DDM get record operation and an equivalent user-written APPC operation is very close, with APPC being slightly quicker. The DDM operation does require more processing seconds than the direct APPC interface.

  Also, because each DDM operation always requires an operation result response from the remote system to ensure data integrity, user-designed partner APPC programs can offer an advantage for update, add, and delete operations by not validating the result of the operation until a later time.
- Be aware that additional time is needed when accessing files on other systems, particularly the time required for communications over the line. This should be considered when determining whether the file should be a local or remote file, especially if it is to be used often.

DDM Conversation Length Considerations

Consider the following information regarding the length of conversations when using DDM:

- Within a source job, if it is likely that a DDM conversation will be used by more than one program or DDM file, *KEEP is the value that should be specified for the DDMCNV job attribute. This saves the time and resources needed to start a target job (TDDM) each time a DDM file is accessed for the same location and mode combination within the source job.
- There is significant system and communications line overhead when a target DDM manager is started. The processing includes the APPC program start request, system type identification, and file open processing. However, if it is not necessary to keep the conversation active, *DROP should be specified for DDMCNV. When the local DDM file is closed, the session being used is released for use by other jobs using DDM or other APPC functions, such as SNADS and display station pass-through, to the same remote system.
- When the source and target systems are AS/400 systems or System/38s, the file input and output requests made by an application program use a form of DDM support that minimizes the amount of time needed to code and decode each request. This is accomplished by System/38 extensions to the DDM architecture.

  When the source and target systems are neither an AS/400 system nor a System/38, then System/38 extensions to the DDM architecture are not used.

Problem Analysis on the Remote System

Some functions that involve a target system may take a relatively long period of time to complete. In these situations, the target system may not appear to be functioning when it is actually waiting for a reply. Any messages created on the target system (such as file full) are sent to the system operator's message queue on the target system. (All DDM-related messages are logged in the target system's job log.) In most cases, a message similar to the one sent to the target system operator is also sent to the source system (with a different message number), but only after the target system operator has replied to the message.

If no job log is found on the target system, the Submit Remote Command (SBMRMTCMD) command can be used to send a Change Job Description (CHGJOBD) command to the target system to change the message logging level.

Another consideration is when end-of-file delay is being used between two AS/400 systems. When this function is being used, canceling the job on the source system does not cancel the job on the target system. Or, if the source system job is canceled while the target job is performing some function, the target job is not canceled.

In some situations, it may be necessary for a user on either the source or target system to call the other location or use pass-through to determine the status of the job on that end and to reply to any messages waiting for a response.

System/36 Source and Target Considerations

Before an AS/400 system can access files on a System/36, Level 1.0 of the DDM architecture (Release 5 or later of System/36 DDM) must be installed on the System/36.

The following sections contain information that applies when an AS/400 system is the source or target system communicating with a System/36. Described are:

- DDM-related differences between AS/400 and System/36 files
- System/36 source to AS/400 target considerations
- AS/400 source to System/36 target considerations

DDM-Related Differences between AS/400 and System/36 Files

Because of differences between the types of files supported by an AS/400 system and a System/36, several items need to be considered when DDM is used between these two systems. Generally, when a System/36 file is created locally (by the BLDFILE utility, for example), the System/36 user specifies such things as the type of file (S = sequential, D = direct, or I = indexed), whether records or blocks are to be allocated, how many of them are to be allocated, and how many additional times this amount can be added to the file to extend it.

Also, you can specify whether the file is to be *delete-capable* (DFILE) or not (NDFILE). In files specified as *not delete-capable*, records can be added or changed in the file, but not deleted.

Once these attributes have been specified, System/36 then creates the file and fills it with the appropriate hexadecimal characters. If a System/36 user specifies the file as:

- A *sequential* file, the entire file space is filled with hex 00 characters and the end-of-file (EOF) pointer is set to the beginning of the initial extent. If you attempt to read an empty sequential file, an EOF condition is received.
- A *direct* file that is *delete-capable*, the entire file space is filled with hex FF characters (deleted records) and the EOF pointer is set to the end of the initial extent. If you attempt to read an empty direct file that is delete-capable, a record-not-found condition is received.
- A *direct* file that is *not delete-capable*, the entire file space is filled with hex 40 characters (blank or null records) and the EOF pointer is set to the end of the initial extent. If you attempt to read an empty direct file that is not delete-capable, a blank record is returned for every record in the file until the end of the file is reached.
- An *indexed* file, it is prepared in the same manner as sequential files.

Typically, once a delete-capable file has been in use, it contains a relatively continuous set of active records with only a few deleted records, possibly an end of data marker, and then a continuous set of deleted records to the end of the file (EOF) space. This means that, unless the file is reorganized, a user can *un*delete (recover) a deleted record.

Of the three types of System/36 files, System/36 indexed files differ little from AS/400-supported logical files. If an AS/400 source program is to use DDM to access the other types of files on a System/36, the AS/400 application programmer should first consider the items remaining in this chapter that relate to System/36.

System/36 Source to AS/400 Target Considerations

When System/36 is using DDM to communicate as a source system to access files on an AS/400 target system, the following information applies and should be considered:

- When System/36 creates a *direct* file on an AS/400 system, the AS/400 system creates a nonkeyed physical file with the maximum number of records, and prepares them as deleted records. The DDM architecture command Clear File (CLRFIL), when issued from a non-AS/400 source system, clears and prepares the file; the CL command Clear Physical File Member (CLRPFM), when issued by a local or remote AS/400 system, does not prepare the file.
- System/36 supports a maximum of three key definitions for logical files and one key definition for keyed physical files.

AS/400 Source to System/36 Target Considerations

When an AS/400 system is using DDM to communicate as a source system to access files on a System/36 target system, the following information applies and should be considered:

- Some file operations that are not rejected by a target AS/400 system may be rejected by a target System/36. Examples are:
  - A delete record operation is rejected if the System/36 file is not a delete-capable file. To the AS/400 source user, the rejection may appear to occur for unknown reasons.

- Change operation that attempts to change the key in the primary index of a System/36 file is always rejected.

- In the System/36 environment, when System/36 users try to copy a delete-capable file to a file that is not delete-capable with the NOREORG parameter, a warning message is issued stating that deleted records may be copied. The user can choose option 0 (Continue) to continue the process. By selecting this option, the file is copied and any deleted records in the input file become active records in the output file. An AS/400 system rejects the copy request if the user specifies COMPRESS(*NO).

- If data is copied to a target System/36 file that is a direct file and is not delete capable, default values for all Copy File (CPYF) command parameters except FROMMBR, TOMBR, and MBROPT must be specified.

- An AS/400 system does not support the overwriting of data on the Delete File (DLTF) command. If an AS/400 user accessing a System/36 wants to overwrite the data, an application program must be written on the AS/400 system, or the user must access the target System/36 and perform the overwrite operation.

- Depending on how a System/36 file is originally created, the maximum number of records it can contain is approximately eight million. This number may be significantly smaller if the file is not extendable or if sufficient storage space is not available to extend the file to add more records.

- System/36 supports a maximum of three key definitions for logical files and one key definition for keyed physical files.

- System/36 file support does not allow a file with active logical files to be cleared. When some AS/400 programs (like COBOL/400 programs) open a file for output only, a command to clear the file is issued. A target System/36 rejects any such command to clear the file if a logical file exists over the file to be cleared.

- System/36 file support automatically skips deleted records. If an AS/400 source user wishes to change the records in a System/36 base file over which at least one logical file has been built, the file must be opened in I/O mode, specifying direct organization and random access by record number. Then each record can be read by record number and changed. If a deleted record is found, a record-not-found indication is returned, and the record may be written rather than rewritten for that position (PL/I write operation rather than a change operation).

- System/36 file support also handles file extensions differently, depending on the file type and the language being used. However, an AS/400 user cannot extend any type of System/36 file unless the access method used to access the file is similar to the method used when the file was created.

  If an AS/400 user is accessing a System/36 file with an access method similar to the one used to create the file, the file can be extended during its use in the following manner:

  - If the file was created as a *sequential* file, the AS/400 user should, if the AS/400 language is:

    - COBOL/400 programming language: open the file using the EXTEND option.
    - PL/I: open the file using the UPDATE option. Perform a read operation using the POSITION option of LAST, and then perform the write operations.

    (BASIC and RPG/400 programming language both handle any needed file extensions automatically.)

  - If the file was created as a *direct* file, the AS/400 user should, if the AS/400 language is:

    - COBOL/400 programming language: open the file using the I-O option, position the end of file pointer to the end of the file (using, for example, READ LAST), and perform a write operation.
    - PL/I: open the file using the UPDATE option, position the end of file (EOF) pointer to the end of the file (using, for example, READ LAST), and perform a write operation.

    (BASIC and RPG/400 programming language both handle any needed file extensions automatically.)

- If the file was created as an *indexed* file, the file is extended each time a write operation is performed for a record having a key that does not already exist in the file.

- The AS/400 user can access sequential System/36 files using either sequential or direct (by relative record number) methods, but significant differences occur when EOF or end of data occurs. If a System/36 sequential file is being processed using relative record number access and is opened for either input/output or output only, then, on reaching the end of active records (EOF), you cannot add new records in the available free space beyond the end of data. You will have to close and reopen the file to extend it. To extend the file, you can either reopen it as a sequential file or open a logical file that uses this file as the base file.

- Because the normal access method used for a System/36 file can be changed by AS/400 parameters to values other than *RMTFILE, it is possible that DDM may attempt to access the System/36 file in ways that the System/36 may not support. Normally, the default value (*RMTFILE) on the ACCMTH parameter gives the user the needed method of access. The use of access methods not normally expected (such as direct or sequential access to indexed files, or sequential access to direct files) requires the use of an ACCMTH parameter explicitly for the access.

  The normal access method used for a System/36 file can be changed on the AS/400 system: by the ACCMTH parameter of the DDM file commands Create DDM File (CRTDDMF) and Change DDM File (CHGDDMF), by the SEQONLY parameter of the Override with Database File (OVRDBF) command, or by using the OVRDBF command to override one DDM file with another DDM file having a different ACCMTH value in effect.

- The AS/400 user can access a System/36 file using a member name if the member name is *FIRST, or in some cases *LAST, or if the file name is the same as the member name.

- Target System/36 DDM cannot support creating logical files with duplicate (nonunique) keys, because the System/36 local data management key sort sends messages to the target system console with options 1 or 3 when duplicate keys are detected. This forces the target system operator either to change the file attributes to allow duplicate keys or to cancel the target data manager.

  Note: Never cancel the target data manager using a SYSLOG HALT.

Override Considerations to System/36

When a file override is issued on the AS/400 system to get records in an logical file on a System/36, the results may be different than expected, because of the difference in how each system deals with keyed files. An AS/400 system uses access paths and logical files, which produce a single view of a file. A System/36 logical file can be considered a list of keys and relative record numbers.

When an AS/400 system accesses a System/36 logical file:

- If you specify a relative record number, you receive the record from the underlying System/36 base file that corresponds to that record number. Then if you request to read the next record, you receive the next sequential record from the base file.

- If you specify a key, you receive the record that corresponds to the first occurrence of that key in the index file. If you request to read the next record, you receive the record that matches the next entry in the index file.

The following example shows the various results for records being retrieved from a System/36 logical file by an AS/400 program. The example assumes that:

- File S36FILEA is the base file and S36FILEB is the logical file that is built over the base file.

- Both files have DDM files named S36FILEA and S36FILEB that point to corresponding remote files on the target System/36.

- The key field is numeric and it always contains the record number.

- The records in the base file (S36FILEA) are in ascending sequence by key, and the records in the logical file (S36FILEB) are in descending sequence with the same key.

- To create the results shown in the following table, the POSITION parameter value is

shown to vary, and no NBRRCDS parameter is specified on either command (which means the total records read is dependent only on the POSITION parameter value).

```
OVRDBF   FILE(S36FILEA)  TOFILE(S36FILEB)
   POSITION(*RRN ... or  *KEY ...)
CPYF   FROMFILE(S36FILEA)  TOFILE(AS400FILEB)
CRTFILE(*YES)  FMTOPT(*NOCHK)
```

Depending on the values specified on the Override with Database File (OVRDBF) command for the POSITION parameter, the following are the resulting records that are copied into the file AS400FILEB when it is created on the source AS/400 system:

POSITION Parameter (See Note)	Resulting Records Retrieved
*RRN 1	299 records, 1 through 299
*KEY 1	1 record, first record only
*RRN 299	1 record, last record only
*KEY 299	299 records, 299 through 1
*RRN 150	150 records, 150 through 299
*KEY 150	150 records, 150 through 1

Note: This column assumes only one key field for *KEY values and uses the remote file name as the default value for the record format name.

Personal Computer Source to AS/400 Target Considerations

The personal computer uses DDM to communicate as a source system to access objects on an AS/400 target in two ways. DDM/PC accesses record-oriented files using Level 1.0 of the DDM architecture function. PC Support/400 uses Level 3.0 of the DDM architecture stream file access support to access folder management services (FMS) folders and documents.

The following considerations apply to PC Support/400 use of the OS/400 DDM target support for the DDM architecture, Level 3.0. Other source systems that send Level 2.0 or Level 3.0 DDM architecture requests for stream files and directories may be able to use this information to help in connecting to an AS/400 system via DDM.

- A FMS must follow the file or directory name to access folder management services (FMS) folders and documents. There can be one or more blanks between the end of the name and the FMS.
- A leading slash (/) signifies the name is fully qualified. If there is no leading slash, any current directory in use is added to the front of the name given.
- The total length of a fully qualified document name is 76 characters. This includes any current directory that may be in use. This does not include the trailing FMS, which is used for typing purposes.
- A / FMS signifies the root folder for a directory name.
- To reduce the number of messages logged to the job log, some errors occurring on the AS/400 target during open, get, put, and close document operations are not logged to the job log. See Figure 6-1 for an illustration of these return codes.

Figure 6-1. AS/400 Return Codes

Description	DDM Reply	Function
Folder not found	DRCNFNRM	OPEN
Folder in use	DRCIUSRM	OPEN
Document in use	FILIUSRM	OPEN
Document not found	FILNFNRM	OPEN
Document not found	EXSCNDRM	DELFIL
Document is read only	ACCINTRM	OPEN
End of data	SUBSTRRM	GET
Data stream (DS) in use	STRIUSRM	GET
Data stream (DS) in use	STRIUSRM	PUT
Substring not valid	SUBSTRRM	UNLOCK
Unlocking a region that is not locked	EXSCNDRM	UNLOCK
File already open for the declare name	OPNCNFRM	OPEN
File not open	FILNOPRM	GET, PUT, LOCK, UNLOCK
Delete document SHDONL(TRUE) specified, but shadow does not exist	EXSCNDRM	DELFIL

- To provide better performance, the AS/400 target handles the closing document in a manner such that when the document is closing, a command completion reply message (CMDCMPRM) is returned to the source system before the document is actually closed. If the document is damaged during the closing time, the user never receives this reply message unless he views the job log. When the user opens the file again, the updated data may not be there.
- An AS/400 system does not support wait on the locking data stream function. The user on the source system must handle the wait function.

Appendix A. Examples of Coding DDM-Related Tasks

The examples in this appendix are based on representative application programs that might be used for processing data both on the local AS/400 system and on one or more remote systems. The first example is a simple inquiry application, and the second example is an order entry application. The third example accesses multiple files on multiple AS/400 systems. The fourth example accesses multiple AS/400 systems and a System/36.

The coding for each of these examples and tasks has one or two parts:

- Coding, shown in pseudo-coded form, not related to DDM but used to build the programming environment. The examples show you the task steps needed, independent of the language you use for your applications. You can write or adapt your programs in your language with the necessary coding to perform these or similar tasks.
- Coding, mostly done in CL, related to communicating with the other systems using DDM in the network.

References are made to other parts of this manual and to other manuals for additional information that is helpful in understanding or using these examples.

Communications Setup for Examples and Tasks

This section describes the network in which DDM is used for the following task examples. The network contains a central system in Philadelphia (an AS/400 system), two remote AS/400 systems in Toronto and New York City, a System/38 in Chicago, and a System/36 in Dallas. The advanced program-to-program communications (APPC) network for these systems was configured with the values shown in Figure A-1 on page A-2.

In this set of task examples, the System/36 has Release 5 of DDM installed and DDM with the compatible PTF installed. The System/38 has Release 8 of CPF installed with the DDM licensed program and the compatible program temporary fix (PTF) change applied to the system.

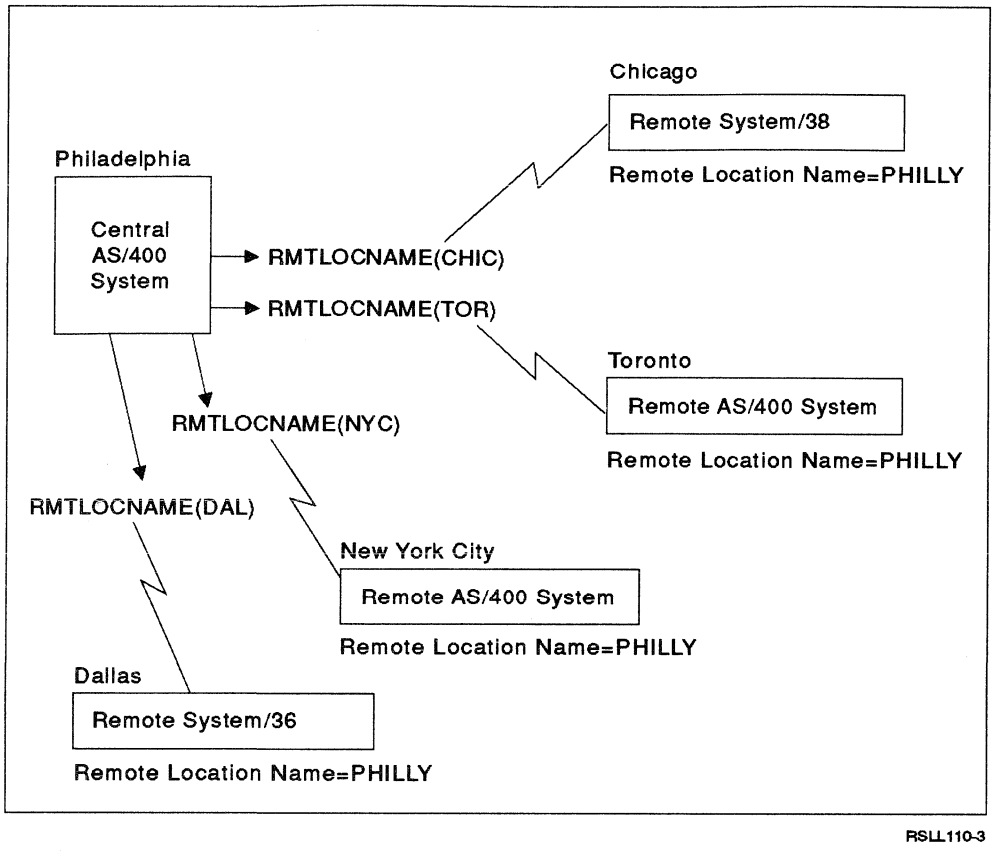


Figure A-1. DDM Network Used in ORDERENT Application Tasks

Example 1. Simple Inquiry Application

This first example shows how multiple locations in a customer's business may be processing the same inquiry application on their own systems, using their own primary files. Without DDM, the two locations shown here (Chicago and Toronto) have their own primary file (CUSTMAST), both with different and duplicate levels of information.

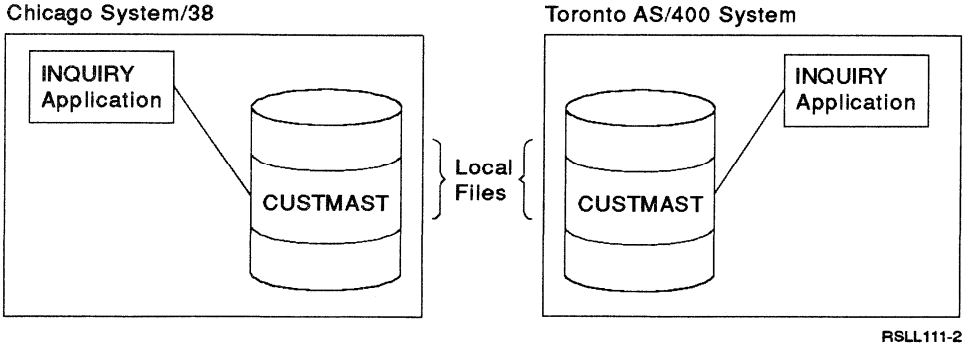


Figure A-2. Two Non-DDM Systems Doing Local Inquiries

The following program (in pseudo-coded form) is run at each location to access its own primary file named CUSTMAST.

```
      Open CUSTMAST
LOOP: Prompt for CUSTNO
      If function 1, go to END
      Get customer record
      Display
      Go to LOOP
END:  Close CUSTMAST
      RETURN
```

Using DDM, the CUSTMAST files are consolidated into one file at a centralized location (Philadelphia, in these examples), and then the local files in Chicago and Toronto can be deleted. The inquiry program used at each remote location and at the central location to access that file is identical to the program used previously.

To perform *remote* inquiries without changing the program, each of the remote locations need only create a DDM file and use an override command:

```
CRTDDMF   FILE(INQ)  RMTFILE(CUSTMAST)  RMTLOCNAME(PHILLY)
  :
OVRDBF  FILE(CUSTMAST) TOFILE(INQ)
```

The DDM file points to the Philadelphia system as the target system and to the CUSTMAST file as the remote file. The same values for this command can be used at each remote location if they also have a remote location named PHILLY. For more information on these parameters, see the Create DDM File (CRTDDMF) command description in the *CL Reference* manual.

Because CUSTMAST is the file name used in the program, the Override with Database File (OVRDBF) command must be used to override the nonexistent CUSTMAST file with the DDM file INQ. (If the CUSTMAST file still exists on the local system, the override is needed to access the central system's primary file; without it, the local file is accessed.)

Figure A-3 shows the same two systems accessing the centralized CUSTMAST file via their DDM files, each named INQ.

An alternative to this approach is to leave the CUSTMAST files on the Chicago and Toronto systems and use them for nonessential inquiries, such as name and address, and use the central CUSTMAST file in Philadelphia for any changes. The CUSTMAST files on the Chicago and Toronto systems could be changed periodically to the current level of the primary file on the Philadelphia system.

This alternative method will be used in the next example.

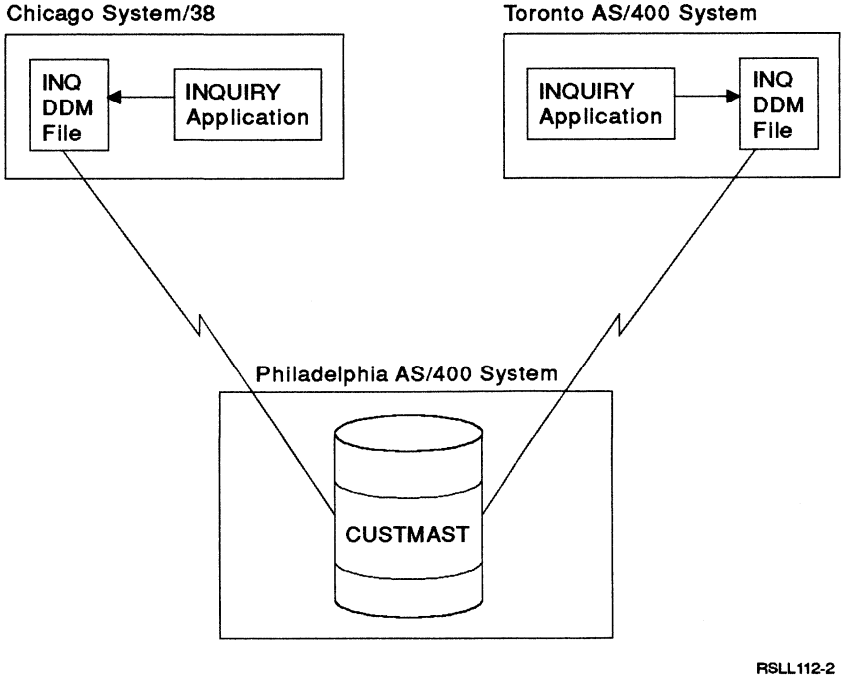


Figure A-3. Two DDM Systems Doing Remote Inquiries

Example 2. ORDERENT Application

This second example shows how multiple locations in a customer's business can process the same order entry application using DDM. The first task in this example shows how to use DDM to put copies of the same application program on remote systems with one primary file at a central location. The second task in this example shows how to use DDM to copy a file to a remote system.

Central System ORDERENT Files

At the central site of Philadelphia, the four files in Figure A-4 are being used by the ORDERENT application program:

At the central system, the CUSTMAST file is a physical file that is the primary file of customer data for all locations. The CUSTMST2 file is a logical file that is based on the CUSTMAST physical file. Using a logical file at the central system provides at least two advantages:

- The same program, ORDERENT, can be used *without* change by the central system and by each of the remote systems.
- The data can be accessed through a separate file and cannot keep a customer's primary record locked for the duration of the order.

The four files at the central site are used as follows:

- The CUSTMAST file contains all the data about all its customers. After a customer order is completed, the CUSTMAST file is changed with all the new information provided by that order.
- The CUSTMST2 file, which is a logical file at the central system, is used at the beginning of a customer order. When an operator enters a customer number, the program reads the customer data from the CUSTMST2 logical file, but the data actually comes from the primary file, CUSTMAST.
- The INVEN file contains the current quantities of all items available for sale to customers. When the operator enters an item number and quantity ordered, the corresponding primary item in the INVEN file is changed.
- The DETAIL file is a list of all the individual items ordered; it contains a record for each item and quantity ordered by customers.

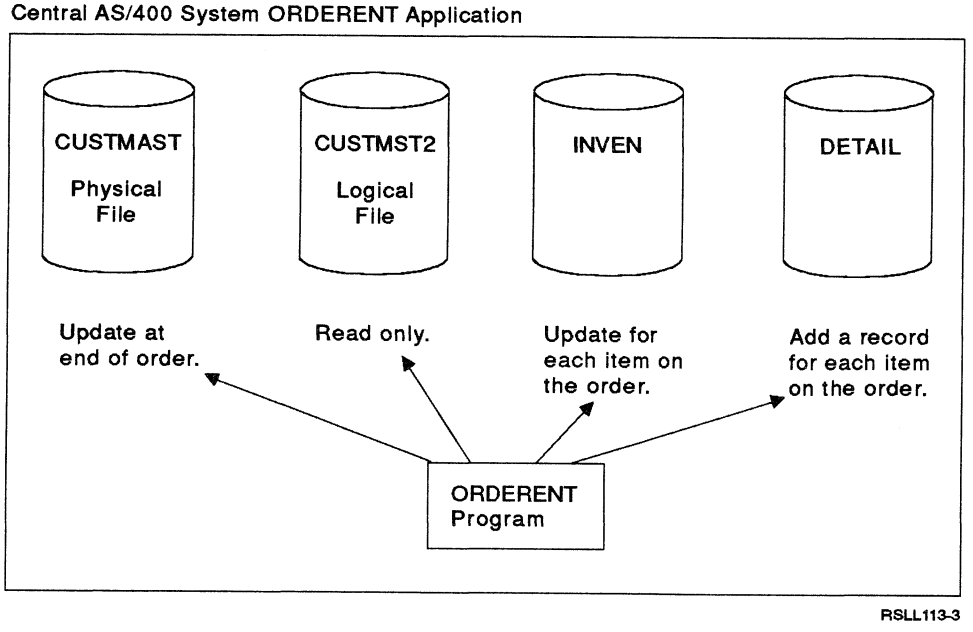


Figure A-4. Files Used by Central System ORDERENT Program

Description of ORDERENT Program

Initially, the ORDERENT program exists only in library PGMLIB on the central system (in Philadelphia). This program does the following:

- When an order entry operator enters a customer number, ORDERENT reads the customer number, then reads the first member of file CUSTMST2 in the PGMLIB library to find the customer name, address, and other information. The retrieved information is displayed to the operator, and the program asks for an item number and quantity desired.
- When the operator enters an item number and quantity desired and presses the Enter key, the program changes the corresponding primary item in the first member of the INVEN file, and it adds a record to the DETAIL file for each item and quantity entered. The program continues asking for another item number and quantity until the operator ends the program.
- When the operator ends the program, the file CUSTMAST is changed with the information for the entire order. (See the pseudo-code of ORDERENT for details.)

For the following examples, it is assumed that all users on the remote systems who need to access CUSTMAST in Philadelphia already have authority to do so, and that those who do not need authority do not have it. In these examples, the AS/400 system in Chicago does not have a compiler.

If we want this program to be used at all the remote locations that also stock a physical inventory, the program needs to be sent to each of the remote systems. We can assume that each of the remote systems has its own inventory and primary files INVEN, DETAIL, and CUSTMST2 (which is a copy of CUSTMAST). How the program can be sent to a remote system is described in "Task 1. Transferring a Program to a Target System" on page A-6.

Pseudo-Code for ORDERENT Program

```
  •
  •
  •
DECLARE CUSTMAST CHANGE
    * Declare file CUSTMAST and allow changing.
DECLARE CUSTMST2 READ
    * Declare file CUSTMST2 as read only.
DECLARE INVEN CHANGE
    * Declare inventory file INVEN and allow changing.
DECLARE DETAIL OUTPUT
    * Declare file DETAIL as output only.
  •
  •
  •
Open CUSTMAST, CUSTMST2, INVEN, and DETAIL files
    * Begin program.
  Show order entry display asking for CUSTNO.
      * Order entry operator enters CUSTNO.
  If function key, go to End.
  Read CUSTNO from display.
    For CUSTNO, return NAME, ADDR, and other
    information from CUSTMST2 file.
  Show NAME, ADDR, and other information on display.
  LOOP: Display 'Item Number ___    Quantity Desired ___'.
      * Order entry operator enters item number and quantity.
    Read ITEMNO and Quantity Desired from display.
    If ITEMNO = 0 then go to LOOPEND.
      Change INVEN with ITEMNO and Quantity Desired.
      Write an item record to the DETAIL file.
    Go to LOOP.
  LOOPEND:  For CUSTNO, change CUSTMAST using
          information in file INVEN.
  End
    * Program has ended.
Close CUSTMAST, CUSTMST2, INVEN, and DETAIL files.
RETURN
```

Figure A-5. Pseudo-Code for ORDERENT Program

Remote Systems ORDERENT Files

The ORDERENT program remains the same at all locations, but the CUSTMST2 file is now a *copy* of the central system's customer primary file CUSTMAST. By using CUSTMST2 whenever possible for data that does not change often, we can minimize the amount of communications time needed to process each order entry request. The remote ORDERENT program reads the local CUSTMST2 file at the beginning of each order, and then, using DDM, updates the CUSTMAST file on the central system only when an order has been completed.

The other two files, INVEN and DETAIL, have the same functions on each remote system as on the central system.

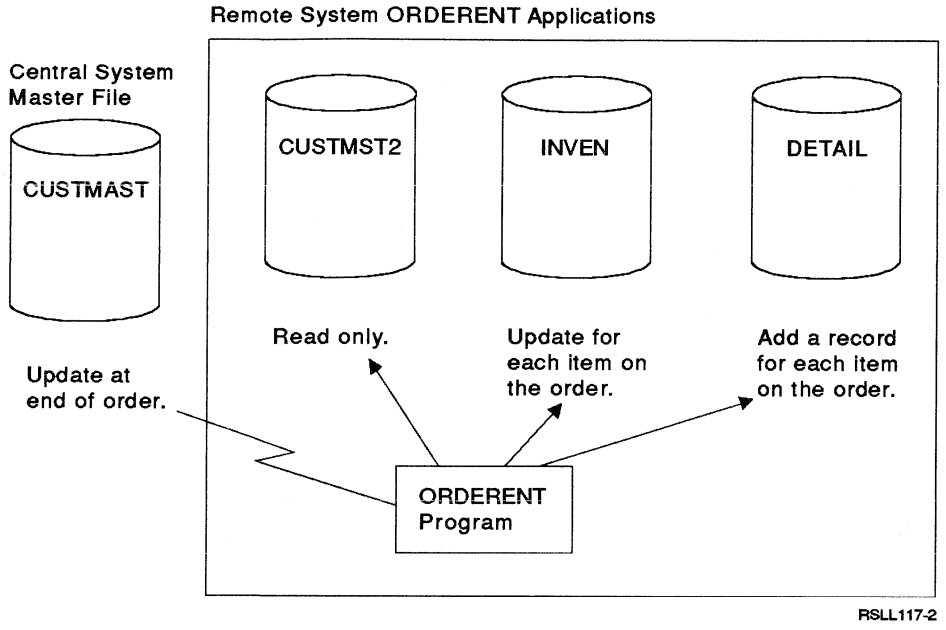


Figure A-6. Files Used by Remote ORDERENT Programs

The CUSTMAST file is changed by all locations and contains the most current information for each customer (for constantly changing data such as the customer's account balance). The CUSTMST2 file, which is used for reading data that changes only occasionally (such as name and address), should be changed periodically (once a week, for example), by recopying the CUSTMAST file into it. Task 2 of this example explains one way to do this.

Task 1. Transferring a Program to a Target System

In this task, the central system in the DDM network, located in Philadelphia, sends a program named ORDERENT to a remote System/38 in Chicago.

The program ORDERENT is transferred from the Philadelphia system to the user in Chicago whose user ID is ANDERSON CHICAGO, and then the program is set up so that ORDERENT in Chicago changes the CUSTMAST file in library PGMLIB on the central system in Philadelphia. The read-only function is performed against the local file (in Chicago) and the change is done in the remote file (in Philadelphia).

For this task, two methods are shown for transferring the ORDERENT program in Philadelphia to the remote system in Chicago. Basically, the same sets of commands are used in both methods, except that the second group of commands used in the pass-through method are embedded in Submit Remote Command (SBMRMTCMD) commands used in the SBMRMTCMD method.

- The first method uses pass-through and object distribution, allowing the operator on the source system to set up both systems without involving the target system operator or to using the SBMRMTCMD command. This method can be used only for AS/400 systems or System/38s.
- The second method uses the SBMRMTCMD command because, in this task, the target system is a System/38. (The SBMRMTCMD command can be used when the target system is an AS/400 system or a System/38.)

Pass-Through Method: One set of commands is entered on the source system, a pass-through session is started with the target system, and a second set of commands is entered on the source system and *run* on the target system.

The following commands are issued on the source system in Philadelphia:

```
CRTSAVF  FILE(TRANSFER)
SAVOBJ   OBJ(ORDERENT)  LIB(PGMLIB)  SAVF(TRANSFER)
   UPDHIST(*NO)  DTACPR(*YES)
SNDNETF  FILE(TRANSFER)  TOUSRID(ANDERSON CHICAGO)
```

Next, a pass-through session is started between the Philadelphia and Chicago systems with the Begin Pass-Through (BGNPASTHR) command. (For more information on the use of this command and pass-through, see the *Remote Work Station Guide*.) The session is used at the source system to enter the following commands, which are run on the target system:

```
CRTSAVF  FILE(RECEIVE)
RCVNETF  FROMFILE(TRANSFER)  TOFILE(RECEIVE)
CRTLIB   LIB(PGMLIB)
RSTOBJ   OBJ(ORDERENT)  SAVLIB(PGMLIB)  SAVF(RECEIVE)
CRTDDMF  FILE(CUSTMAST.PGMLIB)  RMTFILE(*NONSTD 'PGMLIB/CUSTMAST')
   DEVD(PHILLY)
```

These commands create a save file named RECEIVE, into which the TRANSFER file is copied after it is received as a network file from the source system in Philadelphia. A library is created on the Chicago system and the RECEIVE file is restored as the ORDERENT program in the newly created library named PGMLIB. Lastly, a DDM file is created on the Chicago system which allows the Chicago system to access the CUSTMAST file on the Philadelphia system (remote location named PHILLY).

SBMRMTCMD Command Method: Commands needed to accomplish the task are entered at the source system. The source system sends commands that are needed on the target AS/400 system by using the Submit Remote Command (SBMRMTCMD) command between the systems.

The following commands are issued on the source system in Philadelphia to send the ORDERENT program to the target system in Chicago:

```
CRTSAVF  FILE(TRANSFER)
SAVOBJ   OBJ(ORDERENT)  LIB(PGMLIB)  SAVF(TRANSFER)
   UPDHIST(*NO)
SNDNETF  FILE(TRANSFER)  TOUSRID(ANDERSON CHICAGO)
CRTDDMF  FILE(CHICAGO)  RMTFILE(xxxxx)  RMTLOCNAME(CHIC)

SBMRMTCMD  CMD('CRTSAVF  FILE(RECEIVE)')  DDMFILE(CHICAGO)
SBMRMTCMD  CMD('RCVNETF  FROMFILE(TRANSFER)
   TOFILE(RECEIVE)')  DDMFILE(CHICAGO)
SBMRMTCMD  CMD('CRTLIB  LIB(PGMLIB)')  DDMFILE(CHICAGO)
SBMRMTCMD  CMD('RSTOBJ  OBJ(ORDERENT)  SAVLIB(PGMLIB)
   SAVF(RECEIVE)')  DDMFILE(CHICAGO)
SBMRMTCMD  CMD('CRTDDMF  FILE(CUSTMAST.PGMLIB)
   RMTFILE(*NONSTD "PGMLIB/CUSTMAST")  DEVD(PHILLY)')
   DDMFILE(CHICAGO)
```

These commands create a save file named TRANSFER, which saves the ORDERENT program and then sends it as a network file to the target system in Chicago. There, the commands embedded in the SBMRMTCMD command are used to create a save file (named RECEIVE) on the target system, receive the TRANSFER file, and restore it as ORDERENT into the newly created PGMLIB library. Lastly, a DDM file is created on the Chicago system which allows the Chicago system wants to access the CUSTMAST file on the Philadelphia system. The Create DDM File (CRTDDMF) command is in System/38 syntax.

After either of these two methods is used to send the ORDERENT program to, and to create the DDM file on, the Chicago system, the ORDERENT program on that system can be used to access the CUSTMAST file on the Philadelphia system.

Task 2. Copying a File

After performing the first task in Example 2, you decide you want to copy the current level of the CUSTMAST file (in Philadelphia) to the system in Chicago so you can bring the CUSTMST2 file up to date. This example assumes that the CUSTMST2 file already exists in Chicago.

The following commands can be used to copy the CUSTMAST file from the Philadelphia system to the CUSTMST2 file on the Chicago system. (These commands are issued on the system in Philadelphia.)

```
CRTDDMF  FILE(PHILLY/COPYMAST)  RMTFILE(*NONSTD 'CUSTMST2.CHICAGO')
   RMTLOCNAME(CHIC)
CPYF  FROMFILE(PGMLIB/CUSTMAST)  TOFILE(PHILLY/COPYMAST)
   MBROPT(*REPLACE)
```

Note: One might assume that, as an alternative method, you could create a DDM file on the source system, use the SBMRMTCMD command to submit a Create DDM File (CRTDDMF) command to the target system, and then *attempt* to use the newly created target DDM file with another SBMRMTCMD command to perform the copy function back to the original system. However, that method *will not work*, because *an AS/400 system cannot be both a source and target system within the same job.*

Example 3. Accessing Multiple AS/400 Files

Using the same communications environment as in the previous examples, you wish to ask inventory questions of identically named files on the two remote AS/400 systems and the remote System/38. To do so, a program must be written

(shown here in pseudo-code) on the central system that can access the files named LIB/MASTER on the systems in Chicago, in Toronto, and in New York. (In this example, the MASTER files are keyed files, and the first member of each of these files is the one used. Also, data description specifications [DDS] for the MASTER files exist on the central system in Philadelphia.)

The program asks the local order entry operator for an item number (ITEMNO), and returns the quantity-on-hand (QOH) information from the files in Chicago, Toronto, and New York.

The following commands are issued on the system in Philadelphia:

```
CRTDDMF  PGMLIB/CHIFILE  RMTFILE(*NONSTD 'MASTER.LIB')
         RMTLOCNAME(CHIC)
CRTDDMF  PGMLIB/TORFILE  RMTFILE(LIB/MASTER)   RMTLOCNAME(TOR)
CRTDDMF  PGMLIB/NYCFILE  RMTFILE(LIB/MASTER)   RMTLOCNAME(NYC)
```

Following is a sample of the pseudo-code to accomplish the task:

```
DECLARE CHIFILE, TORFILE, NYCFILE INPUT
Open CHIFILE, TORFILE and NYCFILE
LOOP:  Show a display asking for ITEMNO
  Read ITEMNO from the display
    Read record from CHIFILE with the key ITEMNO
    Read record from TORFILE with the key ITEMNO
    Read record from NYCFILE with the key ITEMNO
    Write all QOH values to the display
  If not function key, go to LOOP
Close CHIFILE, TORFILE and NYCFILE
END
```

Figure A-7. Pseudo-Code to Access Multiple AS/400 Files

Before the program is compiled, Override with Database File (OVRDBF) commands can be used to override the three files used in the program with local files that contain the external description formats, identical to the remote files being accessed. Doing so significantly reduces the time required for the compile, since the files on the remote system do not have to be accessed then.

After the program has been compiled correctly, the overrides should be deleted so that the program is able to access the remote files.

An alternative to the use of overrides is to keep the file definitions in a different library. The program could be compiled using the file definitions in that library and then run using the real library.

Example 4. Accessing a File on System/36

The following shows how the pseudo-coded program for the previous task can be changed so a MASTER file on the System/36 in Dallas can be accessed in the same way as the MASTER files on the AS/400 systems and System/38 in Example 3.

Assume that either you have pass-through to the System/36, or that an operator at the System/36 can make changes, if necessary, on the System/36 for you.

The following command is issued on the system in Philadelphia:

```
CRTDDMF  FILE(PGMLIB/DALFILE)  RMTFILE(MASTER)
   RMTLOCNAME(DAL)  ACCMTH(*KEYED)
```

Because the remote file referred to by the DDM file named DALFILE is on a System/36, either of two things must be done:

- The record format of the remote file must be described in the program; that is, it must be a program-described file.
- The program must be compiled with the program referring to a local AS/400 file instead of the System/36 file. This local file must have the same record format name as the DDM file name. Note that the local file need not contain any data records.

For more information about describing a non-AS/400 file, see the non-AS/400 considerations under "Data Description Specifications (DDS) Considerations" on page 5-22.

Following is a sample of the pseudo-code to accomplish the task:

```
DECLARE CHIFILE, TORFILE, NYCFILE, DALFILE INPUT
Open CHIFILE, TORFILE, NYCFILE and DALFILE
LOOP:  Show a display asking for ITEMNO
Read ITEMNO from the display
  Read record from CHIFILE with the key ITEMNO
  Read record from TORFILE with the key ITEMNO
  Read record from NYCFILE with the key ITEMNO
  Read record from DALFILE with the key ITEMNO
  Write all QOH values to the display
  If not function key, go to LOOP
Close CHIFILE, TORFILE, NYCFILE and DALFILE
END
```

Figure A-8. Pseudo-Code to Access a System/36 File

Appendix B. DDM-Related CL Command Summary Charts

This appendix shows summary charts containing most of the control language (CL) commands used with DDM: to determine the DDM job environment, to perform remote file processing (by specifying a DDM file name on a file-related parameter of a CL command), or to perform other actions on a remote system by submitting a CL command to the target system on the Submit Remote Command (SBMRMTCMD) command.

The charts show which commands:

- Are file-related (that operate on file objects)
- Are object-related (that operate on objects other than files, in addition to file objects)
- Can be performed on the source side or on the target side
- Can be affected by file overrides via the Override with Database File (OVRDBF) command
- Are allowed, and have a useful purpose, to be submitted to a target AS/400 system to run (via the SBMRMTCMD command), rather than running on the source system

Notes are included in the charts that can be helpful to the DDM user.

The following describes the kinds of information provided in these charts:

- The first column lists all the CL commands that can be used by DDM: (a) to operate on a remote file identified in a DDM file, or (b) to be submitted on a SBMRMTCMD command using a DDM file.
- In the second column, an F means the command is file related, an O means it is related to OS/400 objects other than files, and a blank means neither of these.
- In the third column, an S means the command operates on objects on the source side, and a T means it operates on objects on the target side. For example, with the create commands that create a file or program using a DDM file as a source file, the T indicates that a source file on the target system is used for the creation; the command runs on the source system and creates a file or program on the source system, but uses a source file on the target system to do it.

  If neither S nor T is shown, the name of a DDM file should *not* be specified on the command; the command should not run on the *source* system as a DDM function. However, the command may be useful when submitted on the SBMRMTCMD command to run on the *target* system (see the last column).
- In the last two columns, an X indicates that the command is valid and useful when used with the command indicated at the top (OVRDBF or SBMRMTCMD) of the column. A blank indicates that the command is not valid.

Generally, when the target system is an AS/400 system or a System/38, any CL command that can be used in either a batch job or batch program can be specified on the SBMRMTCMD command. If a command has a value of *BPGM *and* *EXEC specified for the ALLOW attribute, which you can display by using the Display Command (DSPCMD) command, that command *can* be submitted by the SBMRMTCMD command. (The SBMRMTCMD command uses the QCAEXEC system program to run the submitted commands on the target system.)

Notes:

1. The SBMRMTCMD command can be used to send commands to AS/400 or System/38 target systems only and must be in the syntax of the target system.
2. Although most of the commands listed in this chart can be submitted to a remote system with the SBMRMTCMD command, several can just as easily be run on the source system specifying a DDM file name. These commands are listed in the CL command charts under "Target AS/400-Required File Management Commands" on page 5-21 and "Member-Related Commands" on page 5-21.

Figure B-1 (Page 1 of 3). DDM-Related CL Commands

Command Name	Related to File and/or Object		Affects Objects on Source and/or Target		OVRDBF Command	SBMRMTCMD Command[1]
ADDLFM	F			T[2]		X
ADDPFM	F			T[3]		X
ALCOBJ	F	0	S	T		X
CHGDFUDEF				T		X
CHGDTA				T		
CHGJOB						X
CHGLF	F		S	T		X
CHGLFM	F			T[3]		X
CHGNETA						X
CHGOBJOWN	F	0	S			X
CHGPF	F		S	T		X
CHGPFM	F			T[3]		X
CHGQRYDEF				T		
CHGSRCPF	F		S	T		X
CHKOBJ	F	0	S			X
CLOF	F			T	X	X
CLRPFM	F			T		X
COMMIT	F		S	T		X[11]
CPYF	F		S	T	X	X
CPYFRMDKT	F		S	T	X	X[4]
CPYFRMQRYF	F		S	T	X	X
CPYFRMTAP	F		S	T	X	X[4]
CPYSPLF	F			T		X
CPYSRCF	F		S	T	X	X
CPYTODKT	F		S	T	X	X[4]
CPYTOTAP	F		S	T	X	X[4]
CRTBASPGM				T		X
CRTCBLPGM				T		X
CRTCLPGM				T		X
CRTCMD				T		X
CRTDFUAPP				T		X
CRTDFUDEF				T		
CRTDSPF	F			T		X
CRTDUPOBJ		0	S		X	X
CRTICFF	F			T		X
CRTLF	F		S	T		X
CRTPF	F		S	T	X	X
CRTPLIPGM				T		X
CRTPRTF	F			T		X
CRTPRTIMG				T		X
CRTQRYAPP				T		X
CRTQRYDEF				T		
CRTRPGPGM				T		X
CRTRPTPGM				T		X
CRTSRCPF	F		S	T	X	X

Figure B-1 (Page 2 of 3). DDM-Related CL Commands

Command Name	Related to File and/or Object		Affects Objects on Source and/or Target		OVRDBF Command	SBMRMTCMD Command[1]
CRTTBL				T		X
DCLF	F			T		
DLCOBJ	F	O	S	T		X
DLTDFUAPP						X
DLTF	F		S	T		X
DLTQRYAPP						X
DMPOBJ	F	O	S			X[5]
DMPSYSOBJ		O	S			X[5]
DSNDFUAPP				T		
DSNQRYAPP				T		
DSPDTA				T		
DSPFD	F		S	T		X[5]
DSPFFD	F		S	T		X[5]
DSPNETA						X
DSPOBJAUT	F	O	S			X[5]
DSPOBJD	F	O	S			X[5]
DSPPFM	F			T		
ENDCMTCTL	F		S	T		X[11]
FMTDTA				T		X
GRTOBJAUT	F	O	S			X
INZPFM	F			T[2]		X
MOVOBJ		O	S			X
OPNDBF[6]	F			T	X	X
OPNQRYF	F			T	X	[7]
OVRDBF	F		S			X
POSDBF	F			T		
QRYDTA				T		X
RCVF	F			T		
RCVNETF	F					X
RGZPFM	F			T		X
RMVM	F			T		X
RNMM	F			T		X
RNMOBJ	F	O	S	T[8]		X
ROLLBACK	F		S	T		X[11]
RSTLIB			S			X[9]
RSTOBJ	F	O	S			X[9]
RTVDFUSRC				T		X
RTVQRYSRC				T		X
RVKOBJAUT	F	O	S			X
SAVCHGOBJ		O	S			X[9]
SAVLIB			S			X[9]
SAVOBJ	F	O	S			X[9]

Figure B-1 (Page 3 of 3). DDM-Related CL Commands

Command Name	Related to File and/or Object		Affects Objects on Source and/or Target		OVRDBF Command	SBMRMTCMD Command[1]
SBMDBJOB				T		X
SNDNETF	F			T		X
STRBAS				T		X
STRBASPRC		0		T		X
STRCMTCTL	F		S	T		X[11]
STRDBRDR				T		X
WRKJOB		0	S			X[5]
WRKOBJLCK[10]	F	0				X[5]

Notes:

1 The use of the SBMRMTCMD command is not valid with *any* of the commands in these charts unless the target system is an AS/400 system or a System/38.

2 This member-related command can be used only if the target system is an AS/400 system.

3 This member-related command can be used only if the target system is an AS/400 system or a System/38.

4 These commands require intervention on the target system to load a tape or diskette and they may not produce the results expected.

5 When submitted to the target system, these commands produce output on the target system only; the output is not sent to the source system.

6 OPNDBF command: For more information on commitment control restrictions, see "Commitment Control Support" on page 2-4.

7 OVRDBF command: Although this command works when submitted on the SBMRMTCMD command to a target AS/400 system or a System/38, it is *not* recommended.

8 RNMOBJ command: OBJTYPE*FILE must be specified.

9 When submitted to the target system, these commands require target system resources when tape or diskettes are used to produce the output.

10 WRKOBJLCK command: This command displays any locks on the DDM file, not the remote file.

11 This command will work, but its use is not recommended.

Appendix C. DDM Architecture Code Point Attributes

All DDM architecture words are grouped into classes. Each word in DDM specifies the class to which it belongs with a 2-byte hexadecimal code point. The code point is used to reduce the number of bytes needed to identify the class of a word in main storage and in data streams. The code point specifies the location of the class of the word in the *DDM Architecture: Reference* manual.

When a system message is displayed, a reference is made to a hexadecimal code point. This appendix provides a list of those code points arranged by hexadecimal value.

Figure C-1 (Page 1 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
0001	ASSOCIATION	Name with value association
0002	MINLVL	Minimum level
0003	BIN	Binary integer number
0004	BITDR	A single bit data representation
0005	BITSTRDR	Bit string data representation
0006	BOOLEAN	Truth state
0007	QLFATT	Qualified attribute
0008	CHRDR	A graphic character data representation
0009	CHRSTRDR	Character string data representation
000A	CLASS	Object descriptor
000B	CNSVAL	Constant value
000C	CODPNT	Code point attribute
000D	COLLECTION	Collection object
000E	COMMAND	Command
000F	DATE	Date and time
0011	DFTVAL	Default value attribute
0012	DGTSTRDR	Digit string data representation
0013	DGTDR	Numeric character data representation
0014	NOTE	Note attribute
0015	ENULEN	Enumerated length attribute
0016	ENUVAL	Enumerated value attribute
0017	ERROR	Error severity code
0018	FALSE	False state
0019	HELP	Help text
001A	HEXDR	Hexadecimal number data representation

Figure C-1 (Page 2 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
001B	HEXSTRDR	Hexadecimal string data representation
001C	IGNORABLE	Ignorable value attribute
001D	INDEX	File index
001E	INFO	Information only severity code
001F	LENGTH	Length of value attribute
0020	LETTER	Alphabetic character
0021	MAXLEN	Maximum length attribute
0022	MAXVAL	Maximum value attribute
0023	MENU	Menu
0024	MAGNITUDE	Linearly comparable scalar
0025	MINLEN	Minimum length attribute
0026	MINVAL	Minimum value attribute
0027	NAME	Name
002A	NIL	Nil object
002B	NUMBER	Number
002C	OBJECT	Architected data entity
002D	OPTIONAL	Optional value attribute
002E	PRMDMG	Permanent damage severity code
0031	REPEATABLE	Repeatable variable attribute
0032	REQUIRED	Required value attribute
0033	RESERVED	Reserved value attribute
0034	SCALAR	Scalar object
0036	SPCVAL	Special value attribute
0037	SPRCLS	Superclass
0038	STRING	String
003A	SEVERE	Severe error severity code
003B	TRUE	True state
003C	DATA	Encoded information
003D	WARNING	Warning severity code
003E	ACCDMG	Access damage severity code
003F	SESDMG	Session damage severity code
0040	ENUCLS	Enumerated class attribute
0041	CMDTRG	Command target
0042	BINDR	Binary data representation

Figure C-1 (Page 3 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
0043	BYTDR	An 8-bit value data representation
0044	BYTSTRDR	Byte string data representation
0045	TITLE	A brief description
0046	ATTLST	Attribute list
0047	DEFLST	Definition list
0048	DEFINITION	Definition
0049	INHERITED	Inherited definitions attribute
004A	STSLST	Term status array
004B	ARRAY	Object array
004C	ORDCOL	Ordered collection
004D	ELMCLS	Element of enumerated class attribute
0050	CONSTANT	Constant value
005D	INSTANCE_OF	Instance of
0064	CODPNTDR	Code point data representation
0065	DATDR	Date and time data
0066	NAMDR	Name date
0067	MTLEXC	Mutually exclusive attribute
1001	CLRFIL	Clear file
1002	CLOSE	Close file
1003	CRTAIF	Create alternative index file
1004	CLSDRC	Close directory
1005	FRCBFF	Force buffers
1006	DELFIL	Delete file
1007	GETREC	Get record
1008	INSRECNB	Insert by record number
1009	LSTFAT	List file attributes
100A	GETDRCEN	Get directory entry
100B	LCKFIL	Lock file
100C	SETUPDNB	Set update intent by record number
100D	OPEN	Open file
100E	DELREC	Delete record
100F	MODREC	Modify record
1010	OPNDRC	Open directory
1011	RNMDRC	Rename directory

Figure C-1 (Page 4 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
1013	SETNBR	Set cursor to record number
1014	SETBOF	Set cursor to beginning of file
1015	SETEOF	Set cursor to end of file
1016	SETFRS	Set cursor to first record
1017	SETKEY	Set cursor by key
101B	SETUPDKY	Set update intent by key value
101C	SETLST	Set cursor to last record
101D	SETMNS	Set cursor minus
101E	SETNXT	Set cursor to next record
101F	SETPLS	Set cursor plus
1020	SETPRV	Set cursor to previous record
1023	UNLFIL	Unlock file
1024	INSRECEF	Insert record at end of file
1025	SETKEYLM	Set key limits
1028	CRTDIRF	Create direct file
1029	CRTKEYF	Create keyed file
102A	CRTSEQF	Create sequential file
102C	DCLFIL	Declare file
102D	DELDCL	Delete declared name
102E	LODRECF	Load records into file
1032	INSRECKY	Insert by key value
1036	RNMFIL	Rename file
1037	SETKEYFR	Set cursor to first record in key sequence
1039	SETKEYLS	Set cursor to last record in key sequence
103B	SETKEYNX	Set cursor to next record in key sequence
103C	SETKEYPR	Set cursor to previous record in key sequence
103D	UNLIMPLK	Unlock implicit record lock
1040	ULDRECF	Unload records from file
1041	EXCSAT	Exchange server attributes
1042	SETNXTKE	Set cursor to next record with equal key
1043	CHGFAT	Change file attributes
1044	CRTDRC	Create directory
1045	CRTSTRF	Create stream file
1047	GETSTR	Get stream

Figure C-1 (Page 5 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
1048	LCKSTR	Lock stream
1049	PUTSTR	Put stream
104B	UNLSTR	Unlock stream
104C	LODSTRF	Load stream file
104D	ULDSTRF	Unload stream file
104E	CPYFIL	Copy file
104F	CHGCD	Change current directory
1050	CHGEOF	Change end-of-file
1051	DELDRC	Delete directory
1052	QRYSPC	Query space available
1059	QRYCD	Query current directory
1101	BGNNAM	Beginning search name
1102	FILATTRL	File attribute request list
1103	BASFILNM	Base file name
1104	BYPINA	Bypass inactive record
1105	DELDRCOP	Delete directory option
1108	FILCRTDT	File creation date
1109	CSRDSP	Cursor displacement
110A	RELOPR	Relational operator
110B	EOFNBR	End of file record number
110C	FILEXNSZ	File extent size
110D	FILEXPDT	File expiration date
110E	FILNAM	File name
110F	FILSIZ	File size
1110	FILCLS	File class
1111	DFTRECOP	Default record option
1113	LSTACCDT	Last access date
1114	KEYDEF	Key definition
1115	KEYVAL	Key value
1116	MAXGETCN	Maximum get count
1117	FILMAXEX	File maximum number of extents
1118	PRPSHD	Prepare shadow
1119	OVRDTA	Overwrite data
111A	RECCNT	Record count

Figure C-1 (Page 6 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
111B	DELCP	Deletion capability
111C	RECLEN	Record length
111D	RECNBR	Record number
111E	RECNBRFB	Record number feedback
1122	SHDEXS	Shadow exists
1123	SHDONL	Shadow only
1124	UPDCSR	Update cursor
1125	SHDPRC	Shadow processing
1126	ERRFILNM	Error file name
1128	RTNREC	Return record
1129	STRORD	Stream order
112A	FILPRT	File protected
112B	EOFOFF	End of file offset
112F	KEYHLM	Key high limit
1130	KEYLLM	Key low limit
1132	FILHDD	Hidden file
1133	FILSYS	System file
1134	ACCINTLS	Access intent list
1136	DCLNAM	Declared name
1137	DUPFILOP	Duplicate file option
1139	FILBYTCN	File byte count
113A	FILCHGDT	File change date
113B	FILEXNCN	File extent count
113C	FILINISZ	Initial file size
113D	KEYDUPCP	Duplicate keys capability
113F	PRCCNVCD	Conversational protocol error code
1142	RECLENCL	Record length class
1143	RLSFILLK	Release file lock
1145	RQSFILLK	Requested file lock
1146	UPDINT	Update intent
1147	SRVCLSNM	Server class name
1148	RTNCLS	File retention class
1149	SVRCOD	Severity code
114A	SYNERRCD	Syntax error code

Figure C-1 (Page 7 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
114B	TEXT	Text character string
114C	WAIT	Wait for lock
114D	FILSHR	File sharing
114E	ACCMTHCL	Access method class
114F	NEWFILNM	New file name
1150	BYPDMG	Bypass damaged records
1151	LCKMGRNM	Lock manager name
1152	AGNNAM	Agent name
1153	SRVDGN	Server diagnostic information
1154	ALCINIEX	Allocate initial extent
1155	RTNINA	Return inactive record
1156	ALWINA	Allow cursor to be set to inactive record
1157	MAXOPN	Maximum number of files opened
1159	MAXARNB	Maximum active record number
115A	SRVRLSLV	Server product release level
115B	CSRPOSST	Cursor position status
115C	DTALCKST	Data lock status
115D	SPVNAM	Supervisor name
115E	EXTNAM	External name
115F	HLDCSR	Hold cursor position
1160	KEYVALFB	Key value feedback
1161	ALWMODKY	Allow modified keys
1162	ACCORD	Access order
1163	RLSUPD	Release update intent
1164	KEYDEFCD	Key definition error code
1165	DRCNAM	Directory name
1166	MODCP	File modify capability
1169	STRLEN	Stream length
116A	STRPOS	Position of a stream in a stream file
116B	STRSIZ	Stream file size
116D	SRVNAM	Server name
1174	SPCUNT	Space units
1175	SPCTTL	Total space
117E	SPCAVL	Available space

Figure C-1 (Page 8 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
1183	STROFF	Stream offset
118A	LSTARCDT	Last archived date
118B	RQSSTRLK	Request stream lock
118C	STRLOC	Substream location
118D	CPYNEW	Copy to new file option
118E	CPYOLD	Copy to existing file option
118F	NEWDRCNM	New directory name
1191	GETCP	File get capability
1192	INSCP	File insert capability
1194	FILCHGFL	File change flag
1201	KEYUDIRM	Key update not allowed by different index reply message
1204	DFTRECRM	Default record error
1205	CSRNSARM	Cursor not selecting a record position reply message
1206	DTARECRM	Data record reply message not valid
1207	DUPFILRM	Duplicate file name reply message
1208	DUPKDIRM	Duplicate key different index reply message
1209	DUPKSIRM	Duplicate key same index reply message
120A	DUPRNBRM	Duplicate record number reply message
120B	ENDFILRM	End of file reply message
120C	FILFULRM	File is full reply message
120D	FILIUSRM	File in use reply message
120E	FILNFNRM	File not found reply message
120F	FILSNARM	File space not available reply message
1210	MGRLVLRM	Manager level conflict reply message
1211	FILNOPRM	File not opened reply message
1212	FILNAMRM	File name reply message not valid
1214	SHDEXSRM	Shadow exists reply message
1215	RECLENRM	Record length mismatch reply message
121C	CMDATHRM	Not authorized to command reply message
121E	FILTNARM	File temporarily not available reply message
1220	DCLCNFRM	Declare conflict reply message
1221	DRCTNARM	Directory temporarily not available reply message
1224	RECNBRRM	Record number out of bounds reply message

Figure C-1 (Page 9 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
1225	RECNFNRM	Record not found reply message
122D	KEYLENRM	Key length reply message not valid
1230	ACCATHRM	Not authorized to access method reply message
1231	ACCMTHRM	Access method reply message not valid
1232	AGNPRMRM	Permanent agent error reply message
1233	RSCLMTRM	Resource limits reached reply message
1234	BASNAMRM	Base file name reply message not valid
1237	DRCATHRM	Not authorized to directory reply message
123A	EXSCNDRM	Existing condition reply message
123B	FILATHRM	Not authorized to file reply message
123C	INVRQSRM	Invalid request reply message
123D	KEYDEFRM	Key definition reply message not valid
123F	KEYUSIRM	Key update not allowed by same index reply message
1240	KEYVALRM	Key value reply message not valid
1242	OPNCNFRM	Open conflict error reply message
1243	OPNEXCRM	Open exclusive by same user reply message
1244	OPNMAXRM	Opens at the same time exceed maximum reply message
1245	PRCCNVRM	Conversational protocol error reply message
1249	RECDMGRM	Record damaged reply message
124A	RECIUSRM	Record in use reply message
124B	CMDCMPRM	Command processing completed reply message
124C	SYNTAXRM	Data stream syntax error reply message
124D	UPDCSRRM	Update cursor error reply message
124E	UPDINTRM	No update intent on record reply message
124F	NEWNAMRM	New file name reply message not valid
1250	CMDNSPRM	Command not supported reply message
1251	PRMNSPRM	Parameter not supported reply message
1252	VALNSPRM	Parameter value not supported reply message
1253	OBJNSPRM	Object not supported reply message
1254	CMDCHKRM	Command check reply message
1255	DUPDCLRM	Duplicate declared name reply message
1256	DCLNAMRM	Declared name reply message not valid
1257	DCLNFNRM	Declared name not found reply message

Figure C-1 (Page 10 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
1258	DRCFULRM	Directory full reply message
1259	RECINARM	Record inactive reply message
125A	FILDMGRM	File damaged reply message
125B	LODRECRM	Load records count mismatch reply message
125C	INTATHRM	Not authorized to open intent for named file reply message
125E	CLSDMGRM	File closed with damage reply message
125F	TRGNSPRM	Target not supported reply message
1260	KEYMODRM	Key value modified after cursor was last set reply message
1261	CHGFATRM	Change file attributes rejected reply message
1262	DRCNAMRM	Directory name not valid
1263	DRCNFNRM	Directory not found reply message
1264	STRIUSRM	Stream in use error
1265	SUBSTRRM	Substream reply message not valid
1266	ACCINTRM	Access intent not valid for access method
1267	DRCIUSRM	Directory in use reply message
1268	STRDMGRM	Stream damaged reply message
1269	DRCENTRM	Directory entry reply message not valid
126A	DUPDRCRM	Duplicate directory name
126B	DRCSNARM	Directory space not available
126C	DTAMAPRM	Data mapping error reply message
126E	LODSTRRM	Load stream count mismatch reply message
126F	RECNAVRM	Record not available reply message
1270	DRCNEMRM	Directory not empty reply message
127E	DRCDMGRM	Directory damaged reply message
1282	DRCSUBRM	Directory contains subdirectory reply message
1283	NEWDRNRM	New directory name reply message not valid
1401	ACCMTH	Access method
1402	ACCMTHLS	Access method list
1403	AGENT	Agent
1404	MGRLVLLS	Manager level list
1405	CMBACCAM	Combined access access method
1406	CMBKEYAM	Combined keyed access method
1407	CMBRNBAM	Combined record number access method

Figure C-1 (Page 11 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
1408	CMNMGR	Communications manager
140A	RECCSR	Record cursor
140B	DELAI	Delete access intent
140C	DIRFIL	Direct file
140D	DSSFMT	Data stream structure format
140F	KEYFLDDF	Key field definition
1410	EXTENT	File extent
1411	RECFIL	Record file manager
1413	GETGETLK	Get intent willing to share with get intents at the same time
1414	GETMODLK	Get intent willing to share with modify intents at the same time
1415	GETNONLK	Get intent not willing to share with any users at the same time
1416	GETAI	Get access intent
1417	INSAI	Insert access intent
1418	DCAL3P	Document content architecture level three
1419	DRCAM	Directory access method
141A	DRCCSR	Directory cursor
141B	DRCEMP	Directory empty option
141C	DRPSHD	Drop shadow
141E	KEYFIL	Keyed file
1420	SEQASC	Ascending key sequence
1421	SEQDSC	Descending key sequence
1422	LCKMGR	Lock manager
1423	ALTINDF	Alternative index file
1424	FILAL	File attribute list
1425	MODGETLK	Modify intent willing to share with get intents at the same time
1426	MODMODLK	Modify intent willing to share with modify intents at the same time
1427	MODNONLK	Modify intent not willing to share with any users at the same time
1428	MODAI	Modify access intent
1429	OBJDSS	Object data stream structure
142A	PRMFIL	Permanent file

Figure C-1 (Page 12 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
142B	DFTREC	Default record
142C	PCEXE	PC EXE formatted stream file
142D	RECINA	Inactive record
142E	RECFIX	Fixed length record
142F	RECIVL	Initially varying length record
1430	RECAL	Record attribute list
1431	RECVAR	Varying length record
1432	RELKEYAM	Relative by key access method
1433	RELRNBAM	Relative by record number access method
1434	RNDKEYAM	Random by key access method
1435	RNDRNBAM	Random by record number access method
1436	RPYDSS	DDM reply data stream structure
1437	RPYMSG	Reply message
1438	RQSCRR	Request correlation identifier
1439	RQSDSS	Request data stream structure
143A	BOF	Beginning of file
143B	SEQFIL	Sequential file
143C	SUPERVISOR	Supervisor
143D	SHRRECLK	Share record lock
143E	TMPFIL	Temporary file
143F	EXCRECLK	Exclusive record lock
1440	SECMGR	Security manager
1441	EOF	End of file
1442	MGRLVL	Manager level
1443	EXCSATRD	Server attributes reply data
1444	CMNAPPC	APPC conversational communications manager
1445	KEYAE	Key after or equal to relational operator
1446	KEYAF	Key after operator
1447	KEYEQ	Key equal relational operator
1448	SERVER	Server
1449	DFTSRCIN	Default source initialization
144A	RECORD	Record
144B	KEYBE	Key before or equal to relational operator
144C	KEYBF	Key before operator

Figure C-1 (Page 13 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
144D	FILIND	File index
144E	ALTINDLS	Alternative index list
144F	FILINDEN	File index entry
1450	DCTIND	Dictionary index
1451	DCTINDEN	Dictionary index entry
1452	MGRNAM	Manager name
1453	MGRADR	Manager address
1454	DRCIND	Directory index
1455	DRCINDEN	Directory index entry
1456	MANAGER	Resource manager
1457	DIRECTORY	Directory file
1458	DICTIONARY	Dictionary
1459	DUPFILDO	Duplicate file reply message duplicate option
145A	EXSCNDDO	Existing condition reply message duplicate option
145C	CLRFILDO	Clear file duplicate option
145D	KEYORD	Key order processing
145E	RNBORD	Record number order processing
145F	DFTTRGIN	Default target initialization
1460	DFTINAIN	Default inactive record initialization
1461	DCAFFT	Document content architecture final form text
1462	CPYNCR	Copy with no create option
1463	STRAM	Stream access method
1464	STREAM	Stream
1465	STRFIL	Stream file
1466	CPYDTA	Copy with data option
1467	CPYNDT	Copy with no data option
1468	CURSOR	Access method cursor
1469	STRCSR	Stream cursor
146A	FILE	File manager
1471	DCARFT	Document content architecture revisable form text
1473	MGRLVLN	Manager level number attribute
1479	QRYSPCRD	Query space reply data
1482	CPYAPP	Copy append option
1483	CPYERR	Copy duplicate file error option

Figure C-1 (Page 14 of 14). DDM Architecture Code Points Attributes

Code Point (Hex)	Term	Message Text
1484	CPYRPL	Copy replace option
1485	EXCSTRLK	Exclusive stream lock
1486	SHRSTRLK	Share stream lock
1487	MODSTRLK	Modify stream lock
1488	DRCALL	Delete all files in directory option
1489	DRCANY	Delete any accessible files in directory

Appendix D. DDM Commands and Parameters

This appendix presents the following topics:

- Subsets of DDM architecture supported by OS/400 DDM
  - Supported DDM file models
  - Supported DDM access methods
- Supported DDM commands and parameters
- User profile authority

For additional information on DDM subsets, see the *DDM Architecture: Implementation Planner's Guide* or the *DDM Architecture: Reference* manual.

Note: The abbreviation *KB* appears throughout the tables in this appendix. It represents a quantity of storage equal to 1024 bytes.

Subsets of DDM Architecture Supported by OS/400 DDM

The AS/400 system supports the following subsets of the DDM architecture.

Supported DDM File Models

OS/400 DDM supports the following DDM file models:

Alternate index file (ALTINDF)
Direct file (DIRFIL)
Directory file (DIRECTORY)
Keyed file (KEYFIL)
Sequential file (SEQFIL)
Stream file (STRFIL)

By using the above file models, the AS/400 system supports access to the AS/400 physical and logical files. The following table shows how DDM file models and AS/400 data files correspond.

Figure D-1. AS/400 Data Files

DDM File Model	Corresponding AS/400 Data File
Alternate index file (ALTINDF)	Logical file with one format
Direct file (DIRFIL)	Nonkeyed physical file
Directory file (DIRECTORY)	Folder management services (FMS) folders or data management libraries
Keyed file (KEYFIL)	Keyed physical file
Sequential file (SEQFIL)	Nonkeyed physical file
Stream file (STRFIL)	Folder management services (FMS) document

The following headings discuss each DDM file model and corresponding AS/400 data file.

Alternate Index File (ALTINDF)

OS/400 DDM supports access to a logical file via the DDM alternate index file model. A logical file allows access to the data records stored in a physical file via an alternate index defined over the physical file. Only single format logical files can be accessed through OS/400 DDM. Logical files with select/omit logic can be accessed but records that are inserted may not be retrievable, if they are omitted by the select/omit logic.

Supported Record Classes: An AS/400 alternate index file can have fixed length record (RECFIX) for storage.

Note: OS/400 DDM supports the DDM file transfer commands Load Record File (LODRECFIL) and Unload Record File (ULDRECFIL) for all of the file models except alternate index file. Only fixed record lengths are allowed on an AS/400 system.

Direct File (DIRFIL)

OS/400 DDM supports access to nonkeyed physical files via the DDM direct file model. The support has the following characteristics:

Delete Capabilities: An AS/400 direct file is delete capable or nondelete capable. A nondelete capable file must have an active default record.

Supported Record Classes: An AS/400 direct file can have a fixed length record (RECFIX) for storage.

Note: The AS/400 system does not support the concept of a direct file. OS/400 DDM creates a direct file by creating a nonkeyed physical file and initializing it, with deleted or active default records, to the maximum size requested. No extensions to the file are allowed.

Directory File (DIRECTORY)

OS/400 DDM supports access to a folder management services folder or a data management library via the DDM directory file model. Folders can be created, opened, renamed, closed, or deleted. Libraries can be created, renamed, or deleted.

Keyed File (KEYFIL)

OS/400 DDM supports access to keyed physical files via the DDM keyed file model. The support has the following characteristics:

Supported Record Classes: An AS/400 keyed file can have fixed length record (RECFIX) for storage.

Sequential File (SEQFIL)

The AS/400 system supports access to nonkeyed physical files via the DDM sequential file model. The support has the following characteristics:

Delete Capabilities: The sequential file can be delete or nondelete capable on an AS/400 system.

Supported Record Classes: The sequential file on an AS/400 system can have a fixed length record (RECFIX) for storage.

Stream File (STRFIL)

OS/400 DDM supports access to a folder management services document via the DDM stream file model.

Supported DDM Access Methods

OS/400 DDM supports the following DDM access methods. DDM abbreviations for the access methods are given in parentheses.

- Combined access access method (CMBACCAM)
- Combined keyed access method (CMBKEYAM)
- Combined record number access method (CMBRNBAM)
- Directory access method (DRCAM)
- Random by key access method (RNDKEYAM)
- Random by record number access method (RNDRNBAM)
- Relative by key access method (RELKEYAM)
- Relative by record number access method (RELRNBAM)
- Stream access method (STRAM)

See Figure D-2 for a summary of the access methods that OS/400 DDM supports for each DDM file model. For a description of these access methods, refer to the *DDM Architecture: Implementation Planner's Guide*.

Figure D-2. Supported Access Methods for Each DDM File Model

		DDM File Models					
Term	**Access Method**	**Sequential File**	**Direct File**	**Keyed File**	**Alternate Index File**	**Stream File**	**Directory File**
CMBACCAM	Combined access	N	T	T	N		
CMBKEYAM	Combined keyed			T	T		
CMBRNBAM	Combined record number	T	T	T	N		
DRCAM	Directory						T
RELKEYAM	Relative by key			T	T		
RELRNBAM	Relative by record number	T	T	T	N		
RNDKEYAM	Random by key			T	T		
RNDRNBAM	Random by record number	T	T	T	N		
STRAM	Stream					T	

N = Not supported
T = Target DDM supported
Blank = Not applicable

DDM Commands and Objects

This section describes the DDM command parameters that an AS/400 system supports for each DDM architecture command. For more detailed information about these parameters, see the *DDM Architecture: Reference* manual.

The description of the commands may include:

- Limitations for the use of each command
- Objects that the source system may send to the target system
- Objects that the target system may return to the source system
- DDM parameters that the AS/400 system supports for the command and how the AS/400 system responds to each parameter

DDM Command Parameters

This section lists alphabetically the DDM commands that the AS/400 system supports. Level 1.0, Level 2.0 and Level 3.0 indicate which level of the DDM architecture is supported by the commands.

CHGCD (Change Current Directory) Level 2.0

This command changes the current path. The path is a string of folders. The current path is added to the front of a file or directory name if it does not start with a slash.

This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
AGNNAM	Not applicable	Ignored	
DRCNAM	Not applicable	AS/400 name	Name formats are system defined. The architecture specifies that a directory name length of zero indicates the root directory for the Change Current Directory command. For other commands, a directory name length of zero indicates the *current* directory which may or may not be the root directory at the time the command is issued.

CHGEOF (Change End of File) Level 2.0 and Level 3.0

This command changes the end-of-file mark of a document. The end may be truncated or expanded. A source AS/400 system does not send this command.

Parameter Name	Source	Target
DCLNAM	Not applicable	Program defined
EOFNBR	Not applicable	Supported
EOFOFF	Not applicable	Supported

CHGFAT (Change File Attribute) Level 2.0

This command changes the attributes of a file, document, or folder.

Parameter Name	Stream File	Directory	Sequential, Direct, and Keyed Files	Alternate Index File
DTAFMT	T			
FILCHGDT	T	T	N	N
FILCHGFL	T	N	N	
FILINISZ	N		S, T	
FILEXNSZ	N		S, T	
FILEXPDT			S, T	
FILHDD	T	T	N	N
FILMAXEX	N		S, T	
FILPRT	T	N		
FILSYS	T	T	N	N
DELCP			N	N
GETCP	T		N	
INSCP			N	
MODCP	T		N	
TITLE	T	T	S, T	S, T

Note:

N = Not supported
T = Target DDM supported
S = Source DDM supported
Blank = Not applicable

CLOSE (Close File) Level 1.0 and Level 2.0

This command ends the logical connection between the source system and the data set accessed on the target system. Once the target DDM begins running this command, it must close the data set regardless of the reply message returned.

Parameter Name	Source	Target
DCLNAM	Program defined	Program defined
SHDPRC	Not sent	Supported

Note: Names are implementation defined.

CLRFIL (Clear File) Level 1.0 and Level 2.0

This command clears an existing file and reinitializes it as if it had just been created.

Parameter Name	Source	Target	Notes
FILNAM	Target defined	AS/400 system	Name formats are system defined.
OVRDTA	Not sent	False only	

CLSDRC (Close Directory) Level 2.0

This command closes a folder. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
DCLNAM	Not applicable	Program defined	Names are implementation defined.

CPYFIL (Copy File) Level 2.0

This command copies one document to another document. If the new document does not exist, it may be created. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
ACCORD	Not applicable	Ignored	
BYPDMG	Not applicable	Ignored	
BYPINA	Not applicable	Ignored	
CPYNEW	Not applicable	Supported	CPYNDT only supported parameter value[1].
CPYOLD	Not applicable	Supported	CPYERR only supported parameter value[1].
DCLNAM	Not applicable	Program defined	Names are implementation defined.
FILNAM	Not applicable	AS/400 name	Name formats are system defined.
NEWFILNM	Not applicable	AS/400 name	Name formats are system defined.

[1]All others are rejected with VALNSPRM.

CRTAIF (Create Alternate Index File) Level 1.0 and Level 2.0

This command creates an alternate index file on the target system.

Parameter Name	Source	Target	Notes
BASFILNM	Program defined	AS/400 name	Name formats are system defined.
DUPFILOP	Not sent	Supported	
FILCLS	Not sent	Ignored	Only ALTINDF is valid for CRTAIF command.
FILHDD	Not sent	Ignored	
FILNAM	Program defined	AS/400 name	Name formats are system defined.
FILSYS	Not sent	Ignored	
KEYDEF	Sent	Supported	AS/400 maximum key length is 120.
KEYDUPCP	Sent	Supported	
RTNCLS	Not sent	Supported	Library QTEMP is used for temporaries.
TITLE	Sent	Supported	

CRTDIRF (Create Direct File) Level 1.0 and Level 2.0

This command creates a direct file on the target system.

Parameter Name	Source	Target	Notes
ALCINIEX	Sent	Ignored	
DCLNAM	Not sent	Supported	Names are implementation defined.
DELCP	Sent	Supported	Value must be TRUE unless DFTRECOP (DFTSRCIN) is specified.
DFTREC	Sent	Supported	
DFTRECOP	Sent	Supported	
DUPFILOP	Not sent	Supported	
FILCLS	Not sent	Ignored	Only DIRFIL is valid for CRTDIRF command.
FILEXNSZ	Sent	Supported	AS/400 default is 1,000 records.
FILEXPDT	Sent	Supported	AS/400 default is *NONE.
FILHDD	Not sent	Ignored	
FILINISZ	Sent	Supported	AS/400 default is 10,000 records.
FILMAXEX	Sent	Supported	AS/400 default is 3.
FILNAM	Program defined	AS/400 name	Name formats are system defined.
FILSYS	Not sent	Ignored	
GETCP	Sent	Supported	
INSCP	Sent	Supported	Only TRUE is valid.
MODCP	Sent	Supported	
RECLEN	Sent	Supported	AS/400 maximum record length = 2**15-2.
RECLENCL	Not sent	Supported	Only RECFIX is valid on an AS/400 system.
RTNCLS	Not sent	Supported	Library QTEMP is used for temporaries.
TITLE	Sent	Supported	

CRTDRC (Create Directory) Level 2.0

This command creates folders or libraries on the target system, based on the name received. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
DCLNAM	Not applicable	Program defined	Names are implementation defined.
DRCNAM	Not applicable	AS/400 name	Name formats are system defined.
FILCLS	Not applicable	Ignored	Only DIRECTORY is valid for CRTDRC command.
FILPRT	Not applicable	Supported	FALSE only for libraries.
RTNCLS	Not applicable	PRMFIL only	
TITLE	Not applicable	Supported	

CRTKEYF (Create Keyed File) Level 1.0 and Level 2.0

This command creates a keyed file on the target system.

Parameter Name	Source	Target	Notes
ALCINIEX	Sent	Ignored	
DCLNAM	Not used	Supported	Names are implementation defined.
DELCP	Sent	Supported	
DFTREC	Not sent	Supported	
DFTRECOP	Not sent	Supported	
DUPFILOP	Not sent	Supported	
FILCLS	Not sent	Ignored	Only KEYFIL is valid for CRTKEYF command.
FILEXNSZ	Sent	Supported	AS/400 default is 1,000 records.
FILEXPDT	Sent	Supported	AS/400 default is *NONE.
FILHDD	Not sent	Ignored	
FILINISZ	Sent	Supported	AS/400 default is 10,000 records.
FILMAXEX	Sent	Supported	AS/400 default is 3.
FILNAM	Program defined	AS/400 name	Name formats are system defined.
FILSYS	Not sent	Ignored	
GETCP	Sent	Supported	
INSCP	Sent	Supported	
KEYDEF	Sent	Supported	AS/400 maximum key length is 120.
KEYDUPCP	Sent	Supported	
MODCP	Sent	Supported	
RECLEN	Sent	Supported	AS/400 maximum record length = 2**15-2.
RECLENCL	Not sent	Supported	Only RECFIX is valid on an AS/400 system.
RTNCLS	Not sent	Supported	Library QTEMP is used for temporaries.
TITLE	Sent	Supported	

Note: When a CRTKEYF request is received by an AS/400 target system, the new keyed file reuses deleted records when it is created. If duplicate keys are allowed (KEYDUPCP=TRUE sent), the order of the duplicate keys is not guaranteed.

CRTSEQF (Create Sequential File) Level 1.0 and Level 2.0

This command creates a sequential file on the target system.

Parameter Name	Source	Target	Notes
ALCINIEX	Sent	Ignored	
DCLNAM	Not sent	Supported	Names are implementation defined.
DELCP	Sent	Supported	
DFTREC	Not sent	Supported	
DFTRECOP	Not sent	Supported	
DUPFILOP	Not sent	Supported	
FILCLS	Not sent	Ignored	Only SEQFIL is valid for CRTSEQF command.
FILEXNSZ	Sent	Supported	AS/400 default is 1,000 records.
FILEXPDT	Sent	Supported	AS/400 default is *NONE.
FILHDD	Not sent	Ignored	
FILINISZ	Sent	Supported	AS/400 default is 10,000 records.
FILMAXEX	Sent	Supported	AS/400 default is 3.
FILNAM	Program defined	AS/400 name	Name formats are system defined.
FILSYS	Not sent	Ignored	
GETCP	Sent	Supported	
INSCP	Sent	Supported	
MODCP	Sent	Supported	
RECLEN	Sent	Supported	AS/400 maximum record length = 2**15-2.
RECLENCL	Not sent	Supported	Only RECFIX is valid on an AS/400 system.
RTNCLS	Not sent	Supported	Library QTEMP is used for temporaries.
TITLE	Sent	Supported	

CRTSTRF (Create Stream File) Level 2.0

This command creates a stream file on the target system. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
ALCINIEX	Not applicable	Ignored	
DCLNAM	Not applicable	Program defined	Names are implementation defined.
DTAFMT	Not applicable	Supported	
DUPFILOP	Not applicable	Supported	
FILCLS	Not applicable	Ignored	Only STRFIL is valid for CRTSTRF command.
FILEXNSZ	Not applicable	Ignored	
FILEXPDT	Not applicable	Ignored	
FILHDD	Not applicable	Supported	
FILINISZ	Not applicable	Ignored	
FILMAXEX	Not applicable	Ignored	
FILNAM	Not applicable	AS/400 name	Name formats are system defined.
FILPRT	Not applicable	Supported	
FILSYS	Not applicable	Supported	
GETCP	Not applicable	Supported	
MODCP	Not applicable	Supported	
RTNCLS	Not applicable	Supported	
TITLE	Not applicable	Supported	

DCLFIL (Declare File) Level 1.0 and Level 2.0

This command associates a declared name (DCLNAM) with a collection of object-oriented parameters in the target agent. This collection is stored by the receiving agent for later use. At the time it is received, the command does not affect objects currently opened by the agent. The primary access to the DCLFIL collection is the DCLNAM parameter.

Parameter Name	Source	Target	Notes
AGNNAM	Not sent	Ignored	Only one agent on an AS/400 system.
DCLNAM	Program defined	Program defined	Names are implementation defined.
DRCNAM	Not sent	AS/400 name	Name formats are system defined.
FILEXNSZ	Not sent	Ignored	Create value is used.
FILMAXEX	Not sent	Ignored	Create value is used.
FILNAM	Program defined	AS/400 name	Name formats are system defined.

DELDCL (Delete Declared Name) Level 1.0

This command deletes a declared agent name.

Parameter Name	Source	Target	Notes
AGNNAM	Not sent	Ignored	
DCLNAM	Program defined	Program defined	Names are implementation defined.

DELDRC (Delete Directory) Level 2.0

This command deletes a folder or a library. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
DELDRCOP	Not applicable	DRCEMP or DRCANY	DRCALL not supported.
DRCNAM	Not applicable	AS/400 name	Name formats are system defined. Generic names are not supported.
OVRDTA	Not applicable	FALSE only	

DELFIL (Delete File) Level 1.0 and Level 2.0

This command deletes a file or document.

Parameter Name	Source	Target	Notes
FILNAM	Target defined generics allowed	AS/400 name	Name formats are system defined. Generic names are only allowed for documents.
OVRDTA	Not sent	FALSE only	The AS/400 system does not support overwriting.
SHDONL	Not sent	Supported	FALSE only for files.

DELREC (Delete Record) Level 1.0

This command deletes the record that currently has an update intent placed on it. It does this without affecting the current cursor position.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.

EXCSAT (Exchange Server Attributes) Level 1.0 and Level 2.0

This command exchanges information between servers, such as the server's class name, architectural level of each class of managers it supports, server's product release level, server's external name, and server's name.

Parameter Name	Source	Target
EXTNAM	Sent	Supported
MGRLVLS	Sent	Supported
SPVNAM	Not sent	Ignored
SRVCLSNM	Sent	Supported
SRVNAM	Sent	Supported
SRVRLSLV	Sent	Supported

Reply Objects

The following reply object is returned:

EXCSATRD Server attributes reply data

FILAL (File Attribute List) Level 1.0, Level 2.0, and Level 3.0

This is a list of file attributes that DDM may request on a LSTFAT, OPEN, or GETDRCEN. Some parameters are only valid for specific file types.

Figure D-3 (Page 1 of 2). File Attribute List

Parameter Name	Source	Target	Notes
ACCMTHLS	Requested	Supported	
BASFILNM	Requested	AS/400 name	Name formats are system defined. Qualified name if FILCLS is ALTINDF.
DELCP	Requested	Supported	
DFTREC	Requested	Supported	
DTAFMT	Not requested	Supported	
EOFNBR	Requested	Supported	
EOFOFF	Not requested	Supported	
FILBYTCN	Not requested	Supported	
FILCHGDT	Requested	Supported	
FILCHGFL	Not requested	Supported	
FILCLS	Requested	Supported	
FILCRTDT	Requested	Supported	
FILEXNCN	Requested	Supported	
FILEXNSZ	Requested	Supported	
FILEXPDT	Requested	Supported	
FILHDD	Not requested	Supported	
FILINISZ	Requested	Supported	
FILMAXEX	Requested	Supported	
FILNAM	Requested	Supported	
FILPRT	Not requested	Supported	
FILSIZ	Requested	Supported	
FILSYS	Not requested	Supported	
GETCP	Requested	Supported	
INSCP	Requested	Supported	
KEYDEF	Requested	Supported	
KEYDUPCP	Requested	Supported	
LSTACCDT	Not requested	Not supported	
LSTARCDT	Requested	Supported	
MAXARNB	Requested	Not supported	
MODCP	Requested	Supported	
RECLEN	Requested	Supported	

Figure D-3 (Page 2 of 2). File Attribute List

Parameter Name	Source	Target	Notes
RECLENCL	Requested	Supported	Only RECFIX is allowed on an AS/400 system.
RTNCLS	Not requested	PRMFIL	Unless the library is QTEMP.
SHDEXS	Not requested	Supported	
STRSIZ	Not requested	Supported	
TITLE	Requested	Supported	Maximum length of text is 50 characters for data file, 44 for document or folder.

FRCBFF (Force Buffer) Level 2.0

This command forces the data of the referred object to nonvolatile storage.

Parameter Name	Source	Target	Notes
DCLNAM	Requested	Program defined	Names are implementation defined.

GETDRCEN (Get Directory Entries) Level 2.0

This command gets a list of folders and/or documents. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
BGNNAM	Not applicable	AS/400 name	Name formats are system defined.
DCLNAM	Not applicable	Program defined	Names are implementation defined.
FILATTRL	Not applicable	Supported	
FILCLS	Not applicable	DIRECTORY or STRFIL only	
FILHDD	Not applicable	Supported	
FILSYS	Not applicable	Supported	
MAXGETCN	Not applicable	Supported	
NAME	Not applicable	AS/400 name	Name formats are system defined.

Reply Objects

The following reply object is possible:

FILAL File attribute list

GETREC (Get Record at Cursor) Level 1.0

This command gets and returns the record indicated by the current cursor position.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.
KEYVALFB	Requested	Supported	
RECNBRFB	Requested	Supported	
RTNINA	As required	Supported	Application dependent.
UPDINT	Not sent	Supported	

Reply Objects

The following reply objects are possible:

RECAL	Record attribute list
RECINA	Inactive record (-1 not supported, maximum =2**15-2)
RECORD	Fixed length record (maximum length 2**15-2)

GETSTR (Get Substream) Level 2.0 and Level 3.0

This command gets stream data from a document. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
DCLNAM	Not applicable	Program defined	Names are implementation defined.
STRLEN	Not applicable	Supported	
STROFF	Not applicable	Supported	
STRPOS	Not applicable	Supported	

INSRECEF (Insert at EOF) Level 1.0

This command inserts a record at the end of the file.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.
KEYVALFB	Requested	Supported	
RECCNT	As required	Supported	Application dependent.
RECNBRFB	Requested	Supported	
RLSUPD	Always FALSE	Supported	
UPDCSR	Not sent	Supported	

Command Objects

The following command objects are possible:

RECINA	Inactive record (-1 not supported, maximum = 2**15-2)
RECORD	Fixed length record (maximum length 2**15-2)

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECNBR	Record number

INSRECKY (Insert Record by Key Value) Level 1.0

This command inserts one or more records according to their key values wherever there is available space in the file.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.
RECCNT	As required	Supported	
RECNBRFB	Requested	Supported	
RLSUPD	Always FALSE	Supported	
UPDCSR	Not sent	Supported	

Command Objects

The following command object is possible:

RECORD Fixed length record (maximum length 2**15-2)

Reply Objects

Because the AS/400 system does not support variable length records, only the following reply object is possible:

RECNBR Record number

INSRECNB (Insert Record at Number) Level 1.0

This command inserts one or more records at the position specified by the record number parameter.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.
KEYVALFB	Requested	Supported	
RECCNT	As required	Supported	
RECNBR	Sent	Supported	
UPDCSR	Not sent	Supported	

Command Objects

The following command objects are possible:

RECINA Inactive record (-1 not supported, maximum = 2**15-2)
RECORD Fixed length record (maximum length 2**15-2)

Reply Objects

The following reply object is possible:

KEYVAL	Key value

LCKFIL (Lock File) Level 1.0 and Level 2.0

This command locks the file for subsequent use by the requester.

Parameter Name	Source	Target	Notes
FILNAM	Target name	AS/400 name	Name formats are system defined.
LCKMGRNM	Not used	Ignored	
RQSFILLK	Sent	Supported	
WAIT	Sent	Supported	

LCKSTR (Lock Substream) Level 2.0 and Level 3.0

This command locks a stream file substream. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
DCLNAM	Not applicable	Program defined	Names are implementation defined.
RQSSTRLK	Not applicable	EXCSTRLK and SHRSTRLK only	
STRLOC	Not applicable	Supported	
STROFF	Not applicable	Supported	
WAIT	Not applicable	Supported	The WAIT parameter is neither rejected nor performed.

LODRECF (Load Record File) Level 1.0 and Level 2.0

This command puts a whole record file on the target system.

Parameter Name	Source	Target	Notes
FILNAM	Sent	AS/400 name	Name formats are system defined.

Command Objects

The following command objects are possible:

RECAL	Record attribute list
RECCNT	Record count
RECINA	Inactive record (-1 not supported, maximum = 2**15-2)
RECORD	Fixed length record (maximum length 2**15-2)

LODSTRF (Load Stream File) Level 2.0

This command sends a whole stream file from the source system to the target system. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
FILNAM	Not applicable	AS/400 name	Name formats are system defined.

Command Objects

The following command objects are possible:

STREAM	Stream
STRSIZ	Stream size

LSTFAT (List File Attributes) Level 1.0, Level 2.0, and Level 3.0

This command retrieves selected attributes of a file, document, or folder.

Parameter Name	Source	Target	Notes
FILATTRL	Sent	Supported	
FILNAM	Target name	AS/400 name	Name formats are system defined.
DCLNAM	Not sent	Supported	Names are implementation defined.

Reply Objects

The following reply object is possible:

FILAL	List file attributes reply data

MODREC (Modify Record with Update Intent) Level 1.0

This command changes the record that currently has update intent placed on it without affecting the current cursor position.

Parameter Name	Source	Target	Notes
ALWMODKY	Sent	Supported	
DCLNAM	Sent	Program defined	Names are implementation defined.

Command Objects

The following command object is possible:

RECORD	Fixed length record (maximum length 2**15-2)

OPEN (Open File) Level 1.0 and Level 2.0

This command establishes a logical connection between the using program on the source system and the object on the target system.

Parameter Name	Source	Target	Notes
ACCINTLS	Sent	Supported	
ACCMTHCL	Sent	Supported	
DCLNAM	Program defined	Program defined	Names are implementation defined.
FILATTRL	Not sent	Supported	
FILSHR	Sent	Supported	
PRPSHD	Not sent	Supported for stream files only	

OPNDRC (Open Directory) Level 2.0

This command opens a folder on the target system. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
ACCMTHCL	Not applicable	DRCAM only	
DCLNAM	Not applicable	Program defined	Names are implementation defined.

PUTSTR (Put Substream) Level 2.0 and Level 3.0

This command puts stream data into a document. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
DCLNAM	Not applicable	Program defined	Names are implementation defined.
STROFF	Not applicable	Supported	
STRPOS	Not applicable	Supported	

Command Objects

The following command object is possible:

STREAM Stream

QRYCD (Query Current Directory) Level 2.0

This command returns the current directory. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target
AGNNAM	Not applicable	Ignored

Reply Objects

The following reply object is possible:

DRCNAM Directory name

Note: A directory name length of zero indicates that the root directory is the *current* directory.

QRYSPC (Query Space) Level 2.0

This command returns the amount of space available to a user. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target
AGNNAM	Not applicable	Ignored

Reply Objects

The following reply object is possible:

QRYSPCRD Query space reply data

RNMDRC (Rename Directory) Level 2.0

This command renames a folder or database library, does not support moving folders, and is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
DRCNAM	Not applicable	AS/400 name	Name formats are system defined. Generic names are not allowed.
NEWDRCNM	Not applicable	AS/400 name	Name formats are system defined. Generic names are not allowed.

RNMFIL (Rename File) Level 1.0 and Level 2.0

This command changes the name of an existing database file or document and can also be used for moving documents.

Parameter Name	Source	Target	Notes
FILNAM	Sent	AS/400 name	Name formats are system defined. Generic names are allowed for documents only.
NEWFILNM	Sent	AS/400 name	Name formats are system defined.

SETBOF (Set Cursor to Beginning of File) Level 1.0

This command sets the cursor to the beginning-of-file position of the file.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.

SETEOF (Set Cursor to End of File) Level 1.0

This command sets the cursor to the end-of-file position of the file.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.

SETFRS (Set Cursor to First Record) Level 1.0

This command sets the cursor to the first record of the file.

Parameter Name	Source	Target	Notes
BYPINA	As required	Supported	Application dependent.
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECNBRFB	Requested	Supported	
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECINA	Inactive record (-1 not supported, maximum = 2**15-2)
RECNBR	Record number
RECORD	Record

SETKEY (Set Cursor by Key) Level 1.0

This command positions the cursor based on the key value supplied and the relational operator specified for RELOPR.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVAL	Maximum = 120	Maximum = 120	Maximum key size allowed by an AS/400 system.
KEYVALFB	Requested	Supported	
RECNBRFB	Requested	Supported	
RELOPR	Sent	Supported	
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECNBR	Record number
RECORD	Record

SETKEYFR (Set Cursor to First Record in Key Sequence) Level 1.0

This command sets the cursor to the first record in the key sequence.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECNBRFB	Requested	Supported	
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECNBR	Record number
RECORD	Record

SETKEYLM (Set Key Limits) Level 1.0

This command sets the limits of the key values for subsequent SETKEYNX and SETNXTKE commands. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
DCLNAM	Not applicable	Program defined	Names are implementation defined.
KEYHLM	Not applicable	Supported	Application dependent.
KEYLLM	Not applicable	Supported	Application dependent.

SETKEYLS (Set Cursor to Last Record in Key Sequence) Level 1.0

This command sets the cursor to the last record of the file in key sequence order.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECNBRFB	Requested	Supported	
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECNBR	Record number
RECORD	Record

SETKEYNX (Set Cursor to Next Record in Key Sequence) Level 1.0

This command sets the cursor to the next record of the file in the key sequence order following the record currently indicated by the cursor.

Parameter Name	Source	Target	Notes
BYPDMG	Not sent	Supported	Application dependent.
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECCNT	As required	Supported	Application dependent.
RECNBRFB	Requested	Supported	
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECNBR	Record number
RECORD	Record

SETKEYPR (Set Cursor to Previous Record in Key Sequence) Level 1.0

This command sets the cursor to the previous record of the file in the key sequence order preceding the record currently indicated by the cursor.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECCNT	As required	Supported	Application dependent.
RECNBRFB	Requested	Supported	
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

KEYVAL	Key value
RECAL	Record attribute list
RECNBR	Record number
RECORD	Record

SETLST (Set Cursor to Last Record) Level 1.0

This command sets the cursor to the last record of the file.

Parameter Name	Source	Target	Notes
BYPINA	As required	Supported	Application dependent.
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECNBRFB	Requested	Supported	
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECINA	Inactive record (-1 not supported, maximum = 2**15-2)
RECNBR	Record number
RECORD	Record

SETMNS (Set Cursor Minus) Level 1.0

This command sets the cursor to the record number of the file indicated by the cursor minus the number of record positions specified by CSRDSP.

Parameter Name	Source	Target	Notes
ALWINA	As required	Supported	Application dependent.
CSRDSP	Sent	Supported	Application dependent.
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECNBRFB	Requested	Supported	
RTNINA	As required	Supported	Application dependent.
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECINA	Inactive record (-1 not supported, maximum = 2**15-2)
RECNBR	Record number
RECORD	Record

SETNBR (Set Cursor to Record Number) Level 1.0

This command sets the cursor to the record of the file indicated by the record number specified by RECNBR.

Parameter Name	Source	Target	Notes
ALWINA	As required	Supported	Application dependent.
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECNBR	Sent	Supported	
RTNINA	As required	Supported	Application dependent.
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECINA	Inactive record (-1 not supported, maximum = 2**15-2)
RECORD	Record

SETNXT (Set Cursor to Next Number) Level 1.0

This command sets the cursor to the next record of the file with a record number one greater than the current cursor position.

Parameter Name	Source	Target	Notes
BYPDMG	Not sent	Supported	Application dependent.
BYPINA	As required	Supported	Application dependent.
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECCNT	As required	Supported	Application dependent.
RECNBRFB	As required	Supported	Application dependent.
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECINA	Inactive record (-1 not supported, maximum = 2**15-2)
RECNBR	Record number
RECORD	Record

SETNXTKE (Set Cursor to Next Record in Key Sequence with a Key Equal to Value Specified) Level 1.0

This command positions the cursor to the next record in the key sequence if the key field of that record has a value equal to the value specified in the KEYVAL parameter. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
DCLNAM	Not applicable	Program defined	Names are implementation defined.
HLDCSR	Not applicable	Supported	
KEYVAL	Not applicable	Maximum = 120	Maximum key size allowed by an AS/400 system.
KEYVALFB	Not applicable	Supported	
RECNBRFB	Not applicable	Supported	
RTNREC	Not applicable	Supported	AS/400 system preferred implementation.
UPDINT	Not applicable	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECNBR	Record number
RECORD	Record

SETPLS (Set Cursor Plus) Level 1.0

This command sets the cursor to the record number of the file indicated by the cursor plus the integer number of records specified by CSRDSP.

Parameter Name	Source	Target	Notes
ALWINA	As required	Supported	Application dependent.
CSRDSP	Sent	Supported	Application dependent.
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECNBRFB	Requested	Supported	
RTNINA	As required	Supported	Application dependent.
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECINA	Inactive record (-1 not supported, maximum = 2 **15-2)
RECNBR	Record number
RECORD	Record

SETPRV (Set Cursor to Previous Record) Level 1.0

This command sets the cursor to the record of the file with a record number one less than the current cursor position.

Parameter Name	Source	Target	Notes
BYPINA	As required	Supported	Application dependent.
DCLNAM	Program defined	Program defined	Names are implementation defined.
HLDCSR	Requested	Supported	
KEYVALFB	Requested	Supported	
RECCNT	As required	Supported	Application dependent.
RECNBRFB	Requested	Supported	
RTNREC	Sent	Supported	AS/400 system preferred implementation.
UPDINT	Sent	Supported	AS/400 system preferred implementation.

Reply Objects

The following reply objects are possible:

RECAL	Record attribute list
RECINA	Inactive record (-1 not supported, maximum = 2 **15-2)
RECNBR	Record number
RECORD	Record

SETUPDKY (Set Update Intent by Key Value) Level 1.0

This command places an update intent on the record that has a key value equal to the key value specified by KEYVAL.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.
KEYVAL	Maximum = 120	Maximum = 120	Maximum key size allowed by an AS/400 system.
KEYVALFB	Requested	Supported	
RECNBRFB	Requested	Supported	
RTNREC	Sent	Supported	AS/400 system preferred implementation is RTNREC(FALSE).

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECNBR	Record number
RECORD	Record

SETUPDNB (Set Update Intent by Record Number) Level 1.0

This command places an update intent on the record of the file indicated by the record number specified by RECNBR.

Parameter Name	Source	Target	Notes
ALWINA	As required	Supported	Application dependent.
DCLNAM	Program defined	Program defined	Names are implementation defined.
KEYVALFB	Requested	Supported	
RECNBR	Sent	Supported	
RTNINA	As required	Supported	Application dependent.
RTNREC	Sent	Supported	AS/400 system preferred implementation is RTNREC(FALSE).

Reply Objects

The following reply objects are possible:

KEYVAL	Key value
RECAL	Record attribute list
RECINA	Inactive record (-1 not supported, maximum = 2 **15-2)
RECORD	Record

ULDRECF (Unload Record File) Level 1.0

This command sends records from a target record file to the source.

Parameter Name	Source	Target	Notes
ACCDORD	Sent	Supported	Application dependent.
BYPDMG	Sent	Supported	Application dependent.
FILNAM	Sent	AS/400 name	Name formats are system defined.
RTNINA	Sent	Supported	Application dependent.

Reply Objects

The following reply objects are possible:

RECAL	Record attribute list
RECCNT	Record count
RECINA	Inactive record (-1 not supported, maximum = 2 **15-2)
RECORD	Record

ULDSTRF (Unload Stream File) Level 2.0

This command sends a document from the target to the source. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
BYPDMG	Not applicable	FALSE only	
FILNAM	Not applicable	AS/400 name	Name formats are system defined.
STRORD	Not applicable	Supported	

Reply Objects

The following reply objects are possible:

STREAM	Stream
STRPOS	Stream position
STRSIZ	Stream size

UNLFIL (Unlock File) Level 1.0 and Level 2.0

This command releases explicit file locks held by the requester on the file.

Parameter Name	Source	Target	Notes
FILNAM	Target name	AS/400 name	Name formats are system defined.
LCKMGRNM	Not sent	Ignored	
RLSFILLK	Sent	Supported	

UNLIMPLK (Unlock Implicit Record Lock) Level 1.0

This command releases all implicit record locks currently held by the cursor.

Parameter Name	Source	Target	Notes
DCLNAM	Program defined	Program defined	Names are implementation defined.

UNLSTR (Unlock Substreams) Level 2.0 and Level 3.0

This command unlocks a stream file substream. This command is not sent by a source AS/400 system.

Parameter Name	Source	Target	Notes
DCLNAM	Not applicable	Program defined	Names are implementation defined.
STRLOC	Not applicable	Supported	

User Profile Authority

The user profile associated with target AS/400 system jobs must be authorized to the equivalent CL commands before the DDM command can be processed. The target job's user profile must be authorized to use the CL commands listed in the following table before DDM requests can be processed.

Figure D-4 (Page 1 of 2). User Profile Authority CL Commands

DDM Command Received	DDM Command Description	Object Type	Authorized CL Command
CHGDRC	Change Current Directory	FLR	NONE [1]
CHGFAT	Change File Attributes	PFILE LF DOC/FLR	CHGPF CHGLF NONE [1]
CLOSE	Close File	FILE DOC	NONE [2] NONE [1]
CLRFIL	Clear File	FILE DOC	NONE NONE [1]
CLSDRC	Close Directory	FLR	NONE [1]
CPYFIL	Copy File	DOC	NONE [1]
CRTAIF	Create Alternate Index File	LF	CRTLF
CRTDIRF	Create Direct File	PF	CRTPF
CRTKEYF	Create Key File	PF	CRTPF
CRTSEQF	Create Sequential File	PF	CRTPF
CRTSTRF	Create Stream File	DOC	NONE [1]
CRTDRC	Create Directory	LIB FLR	CRTLIB CRTFLR
DELFIL	Delete File	FILE DOC	DLTF NONE [1]
DELDRC	Delete Directory	LIB FLR	DLTLIB NONE [1]
GETDRCEN	Get Directory Entry	DOC/FLR	NONE [1]
LCKFIL	Lock File	FILE	ALCOBJ
LODRECF	Load (Put) Records to File	FILE	NONE [3]
LSTFAT	List File Attributes	FILE DOC/FLR	NONE [4] NONE [1]
OPEN	Open File	FILE DOC	NONE [2] NONE [1]
OPENDRC	Open Directory	FLR	NONE [1]
QRYSPC	Query Space Available to User	USRPRF	NONE [5]
RNMDRC	Rename Directory	FLR LIB	NONE [1] RNMOBJ

Figure D-4 (Page 2 of 2). User Profile Authority CL Commands

DDM Command Received	DDM Command Description	Object Type	Authorized CL Command
RNMFIL	Rename File	FILE	RNMOBJ
		DOC	NONE [1]
		MBR	RNMM
UNLFIL	Unlock File	FILE	NONE [6]
ULDRECF	Unload Records From File	FILE	NONE [3]

Notes:

[1] With the exception of CRTFLR, authorization to the CL commands that operate on folders and documents is not verified because there are other ways for the user to start those functions through the OfficeVision/400 interface. If a user is enrolled in OfficeVision/400 (which DDM does verify), that user is allowed to perform all document and folder operations except CRTFLR.

[2] Authorization to a command is not verified because there are means other than using a command interface by which AS/400 users can open and close files.

[3] Command authorization is not verified because there is not a direct, one-to-one mapping between a CL command and the DDM LODRECF/ULDRECF command.

[4] Authorization to the DSPFD and DSPFFD commands is not verified because it cannot be determined which command should be verified. In addition, the conditions under which the DDM command was issued by the source system are not known.

[5] The space available to a user can be obtained by issuing the DSPUSRPRF command, but this is only a small piece of the data available through the use of this command.

[6] Authorization to the CL DLCOBJ command is not checked because if the remote user was able to allocate files, DDM must be able to deallocate them.

The following list is an explanation of the object type codes used in Figure D-4 on page D-37:

Object Type	Object Type Definition
DOC	Document
FLR	Folder
PF	Physical File
LF	Logical File
LIB	Library
MBR	Member
SRCF	Source Physical File
USRPRF	User Profile

Appendix E. AS/400 System-to-CICS Considerations

This appendix discusses AS/400 system to Customer Information Control System (CICS) additional programming considerations.

Note: A System/370* host must have installed CICS/OS/VS Version 1.7 or later and CICS/DDM Version 1.1.

AS/400 Languages, Utilities, and Licensed Programs

The following AS/400 languages, utilities, and licensed programs can access remote CICS files:

- Programs written in the following languages on an AS/400 system can access remote CICS files:

  C/400 Programming Language
  See "C/400 Considerations" on page E-7.

  CL See "AS/400 CL Considerations" on page E-2.

  COBOL/400 Programming Language
  See "COBOL/400 Considerations" on page E-5.

  PL/I See "PL/I Considerations" on page E-4.

  RPG/400 Programming Language
  See "RPG/400 Considerations" on page E-8.

- Programs written in BASIC may cause results that cannot be predicted when accessing remote CICS files.
- AS/400 Query can access the remote entry sequence data set (ESDS), relative record data set (RRDS), and key sequence data set (KSDS). However, AS/400 Query cannot access virtual storage access method (VSAM) files through DDM.
- The following licensed programs may cause results that cannot be predicted when accessing remote CICS files:
  - OfficeVision/400
  - PC Support/400

Note: Some of the high-level languages provide access to the system database I/O feedback area. When accessing a remote VSAM RRDS, this area will contain the relative record number. However, when accessing other types of VSAM data sets, the relative record number is not known and a value of -1 is returned as the relative record number.

Additional considerations may apply when accessing CICS files which are to be read or written by a System/370 host due to the way a System/370 host stores data. For example, the System/370 host representation of a floating point number is different from the AS/400 system representation of a floating point number.

CRTDDMF (Create DDM File) Considerations

For applications running on an AS/400 system to access remote files, the programmer must use the CRTDDMF command to create an object called a DDM file. The ACCMTH parameter of this command shows which DDM access method should be used when opening the remote file. If *RMTFILE is used, OS/400 DDM selects an access method that is compatible with:

- The type of VSAM data set being accessed
- The access methods supported by CICS/DDM for the VSAM data set

Figure E-1 on page E-2 shows how the possible values for the ACCMTH parameter correspond to VSAM data sets.

Figure E-1. ACCMTH Parameter of AS/400 CRTDDMF Command

ACCMTH Parameter Values	VSAM Data Set Organization			
	ESDS	RRDS	KSDS	VSAM Path
*ARRIVAL	R	R	E	E
*KEYED	E	E	R	R
*BOTH	E	O	O	O
*RANDOM	E	O	O	O
*SEQUENTIAL	R	O	O	O
*COMBINED	E	O	E	E

Where:

R Shows the parameter is required for accessing the VSAM data set.

O Shows the parameter is optional for accessing the VSAM data set.

E Shows the parameter causes an OS/400 message.

To improve performance, the AS/400 user may want to supply values other than *RMTFILE for the ACCMTH parameter. To avoid system messages, use the values specified in Figure E-1 when accessing remote VSAM data sets.

The value specified for the RMTFILE file parameter must be the same as the value specified for the DATASET parameter of the CICS DFHFCT macro when the VSAM data set is defined to the CICS system. See the *CICS/OS/VS Resource Definition (Macro)* manual for information about the DFHFCT macro.

AS/400 CL Considerations

Besides the information in this manual, consider the following sections when using AS/400 CL commands to access VSAM data sets on a remote CICS system.

Note: Commands that do not appear in the following headings have no considerations besides those stated in this manual.

ALCOBJ (Allocate Object): Unless the CICS system programmer replaces the CICS/DDM-supplied exclusive file lock program with a special version of the program, a lock condition value of *SHRRD or *SHRUPD must be used when using the Allocate Object (ALCOBJ) command to allocate a remote VSAM data set. All other lock condition values result in a system message on the AS/400 system.

CLRPFM (Clear Physical File Member): The Clear Physical File Member (CLRPFM) command cannot clear a VSAM data set on a remote CICS system.

CPYF (Copy File): The Copy File (CPYF) command can access remote VSAM data sets defined to a CICS system. However, consider the following:

- When the TOFILE parameter is a remote VSAM data set:
  - The CRTFILE parameter must have a value of *NO.
  - The MBROPT parameter must have a value of *ADD.
  - The FMTOPT parameter must have a value of *NOCHK.
- When the TOFILE parameter is a remote VSAM ESDS or KSDS, the COMPRESS parameter must have a value of *YES.

CPYTODKT, CPYFRMDKT, CPYTOTAP, CPYFRMTAP and CPYSPLF Commands: The Copy to Diskette (CPYTODKT), the Copy from Diskette (CPYFRMDKT), the Copy to Tape (CPYTOTAP), and the Copy from Tape (CPYFRMTAP) commands access remote VSAM data sets defined to a CICS system. However, *ADD must be used for the MBROPT parameter. The Copy Spool File (CPYSPLF) command cannot access remote VSAM data sets.

DLCOBJ (Deallocate Object): The Deallocate Object (DLCOBJ) command can release any lock successfully acquired for a remote VSAM data set.

DSPFD and DSPFFD Commands: The Display File Description (DSPFD) and Display File Field Description (DSPFFD) commands can display information about a remote VSAM data set. However, the information for many of the fields is not available to CICS/DDM and is not returned to OS/400 DDM. Refer to Figure E-2 on page E-3 for the fields that CICS/DDM does not return:

Figure E-2. CICS/DDM File and File Field Descriptions

Field	Value
Creation date	Zeros
Current number of records	Zeros
Data space in bytes	Zeros
File level identifier	Zeros
File text description	Zeros
Format level identifier	Blanks
Last change date and time	Zeros
Member creation date	Zeros
Member expiration date	*NONE
Member level identifier	Zeros
Member size	*NOMAX
Number of deleted records	Zeros
Text description	Blanks
Total deleted records	Zeros
Total member size	Zeros
Total records	Zeros

Note: The values displayed do not represent actual data about the file. They act as default values for the information that CICS/DDM does not return.

When the type of file is logical, the information displayed shows that unique keys are not required. In fact, CICS/DDM does not know whether unique keys are required or not.

Sometimes, the AS/400 user needs to know the type of VSAM data set being accessed. By using the following information, which is displayed by the DSPFD command, it is possible for the AS/400 user to make this decision:

- If the type of file is logical, the VSAM data set is a VSAM path.
- If the type of file is physical and the access path is keyed, the VSAM data set is KSDS.
- In all other cases, the VSAM data set is either an RRDS or an ESDS. If the AS/400 user must know whether it is an RRDS or an ESDS, the CICS system programmer should be contacted.

DSPPFM (Display Physical File Member): The Display Physical File Member (DSPPFM) command can be used to access remote RRDS. It does not work for other types of VSAM data sets.

OPNDBF (Open Database File): The Open Database File (OPNDBF) command can open a remote VSAM data set. However, a system message is created on the AS/400 system if *ARRIVAL is used for the ACCPTH parameter and the remote data set is a VSAM KSDS or a VSAM path.

OVRDBF (Override with Database File): The Override with Database File (OVRDBF) command can override a local database file with a remote VSAM data set. However, the following must be considered:

- A POSITION value of *RRN works when the remote VSAM data set is an RRDS. For all other types of VSAM data sets, *RRN causes a system message on the AS/400 system.
- A POSITION value of *KEYB or *KEYBE causes a system message on the AS/400 system when the remote file is a VSAM path.
- Unless the CICS system programmer replaces the CICS/DDM-supplied exclusive file lock program, the RCDDMTLCK parameter must have a lock condition value of *SHRRD or *SHRUPD. All other lock condition values result in a system message on the AS/400 system.
- CICS/DDM does not return the actual expiration date of the file to OS/400 DDM. Instead, it returns a special value that indicates the expiration date is not known. This is true even if the EXPCHK parameter has a value of *YES.

RCVNETF (Receive Network File): The Receive Network File (RCVNETF) command can access a VSAM data set defined to a remote CICS system. However, the MBROPT parameter must have a value of *ADD.

Language Considerations for AS/400 System and CICS

AS/400 application programmers using PL/I, COBOL/400, C/400, AS/400 System/36-Compatible RPG II, or RPG/400 languages should be aware of the information in the following sections.

PL/I Considerations

The following sections summarize the limitations that exist when using PL/I to access remote VSAM data sets from an AS/400 system. These limitations should be considered in addition to those already stated in this manual.

PL/I Open File Requests: The OS/400 DDM user can access remote CICS files with PL/I programs. When opening the file with the RECORD file attribute, the program must use the file attributes specified in Figure E-3. By noting the values that appear in this table, you can determine the difference between accessing an AS/400 database file and a remote VSAM data set.

Note: Remote files can also be opened with the PL/I STREAM file attribute. However, if the STREAM file attribute is used to open a VSAM KSDS, a system message occurs. This happens because records in a VSAM KSDS cannot be processed in arrival sequence.

Unless the CICS system has replaced the CICS/DDM exclusive file locking program, you cannot use the EXCL and EXCLRD file locking options for the ENVIRONMENT parameter when opening a remote VSAM data set.

Figure E-3. PL/I File Attributes

PL/I File Attributes	VSAM Data Set Organization			
	ESDS	RRDS	KSDS	VSAM Path
SEQUENTIAL	R	O	O	O
DIRECT	E	O	O	O
SEQL KEYED	E	O	O	O
INPUT	O	O	O	O
OUTPUT	O	O	O	E
UPDATE	O	O	E	E
CONSECUTIVE	R	R	E	E
INDEXED	–	–	R	R

Where:

R Shows the attribute is required for accessing the VSAM data set.

O Shows the attribute is optional for accessing the VSAM data set.

E Shows the attribute is allowed by PL/I but the open fails when accessing the VSAM data set.

– Shows the option is valid for keyed files only.

PL/I Input/Output Requests: The PL/I programmer must be aware of the following when using the PL/I input/output requests:

Read Requests

- A KEYSEARCH parameter value of BEFORE or EQLBFR is not supported by CICS/DDM when accessing a VSAM path. However, these parameter values are supported when accessing a VSAM KSDS.
- A POSITION parameter value of PREVIOUS and LAST is not supported when accessing a VSAM path. However, these parameter values are supported when accessing a VSAM KSDS.
- Because the DIRECT or SEQUENTIAL KEYED attributes cannot be used to open a VSAM ESDS, it is not possible to access records by relative record number or to have the relative record number returned via the KEY and KEYTO parameters. See "PL/I Open File Requests" on page E-4 for further information.
- Because VSAM KSDS and VSAM alternate indexes are always defined as a single key field, the NBRKEYFLDS parameter should not be used.

Write Requests

- The KEYFROM parameter does not work when writing a record to a VSAM RRDS.
- The WRITE request does not work when writing a record to VSAM KSDS that already contains a record with the same key value.
- Because the OUTPUT or UPDATE file attribute cannot be used to open a VSAM path, it is not possible to write records to a VSAM path. Instead, the application program must write the record by using the base data set of the VSAM path.

Rewrite Requests

- The REWRITE does not work if rewriting a record of a VSAM KSDS when the key value of the record is changed.
- Because the UPDATE file attribute cannot be used to open a VSAM path, it is not possible to rewrite records in a VSAM path. Instead, the application program must rewrite the record by using the base data set of the VSAM path.

Delete Requests

- The DELETE does not work when deleting the record of a VSAM ESDS.
- Because the UPDATE file attribute cannot be used to open a VSAM path, it is not possible to delete records in a VSAM path. Instead, the application program must delete the records by using the base data set of the VSAM path. However, if the base data set of the VSAM path is a VSAM ESDS, the DELETE does not work.

COBOL/400 Considerations

The following sections summarize the limitations that exist when using COBOL/400 programming language to access remote VSAM data sets from an AS/400 system. Unless otherwise stated, these limitations apply to both the System/36 and the AS/400 system. These limitations are in addition to those already stated in this manual.

COBOL/400 Select Clause: AS/400 users can access remote CICS files with COBOL/400 programming language. However, the COBOL/400 SELECT clause must use the file organizations and access methods specified in Figure E-4 on page E-5.

Figure E-4. COBOL/400 File Organizations and Access Methods

COBOL/400 Programming Language		VSAM Data Set Organization			
Program-Given Organization	**Program-Given Access Methods**	**ESDS**	**RRDS**	**KSDS**	**VSAM Path**
Sequential	Sequential	X	X	E	E
Relative	Sequential Random Dynamic	E E E	X X X	E E E	E E E
Indexed	Sequential Random Dynamic	– – –	– – –	X X X	X X X

Where:

X Shows the access method is allowed.

E Shows that COBOL/400 programming language allows the access method but that the open fails when accessing the VSAM data set. An AS/400 message is created.

– Shows the option is never valid for nonkeyed files. An AS/400 message occurs whenever indexed file organization is selected for any nonkeyed file. This is true even when the file is a local file.

Notes:

1. When accessing a VSAM path, the WITH DUPLICATE phrase should be used.
2. When accessing a VSAM KSDS, the WITH DUPLICATE phrase should not be used.

COBOL/400 Statements: The following headings describe considerations you should be aware of when using COBOL/400 statements to access remote VSAM data sets.

COBOL/400 OPEN Statement: When accessing remote CICS files, the COBOL/400 OPEN statement must use the open modes that are specified in Figure E-5.

Figure E-5. Using COBOL/400 Programming Language to Open a CICS File

COBOL/400 Open Mode	VSAM Data Set Organization			
	ESDS	**RRDS**	**KSDS**	**VSAM Path**
Input	X	X	X	X
Output	E	E	E	E
Input/Output	X	X	X	E
Extend	X	–	–	–

Where:

X Shows the open mode is allowed.

E Shows the open mode is allowed by COBOL/400 programming language but that the open fails when accessing the VSAM data set. A message occurs on an AS/400 system.

– Shows the open mode is not applicable.

COBOL/400 READ Statement

- The PRIOR phrase and the LAST phrase are not supported by CICS/DDM when accessing a VSAM path. It is supported when accessing a VSAM KSDS.
- Because the RELATIVE file organization can only be used to open a VSAM RRDS, it is not possible to access records by relative record number or to have the relative record number returned from the remote file unless the remote file is a VSAM RRDS.

COBOL/400 WRITE Statement

- The WRITE statement does not work when the COBOL/400 program is running on an AS/400 system and the file was opened with a RELATIVE file organization.
- The WRITE statement does not work when writing a record to a VSAM KSDS and the data set already contains a record with the same key value.
- Because the input/output and output open modes cannot be used to open a VSAM path, it is not possible to write records to a VSAM path. Instead, the application program must write the record by using the base data set of the VSAM path.

COBOL/400 REWRITE Statement

- The REWRITE statement does not work when rewriting the record of a VSAM KSDS and the key value of the record is changed.
- Because the input/output open mode cannot be used to open a VSAM path, it is not possible to rewrite records in a VSAM path. Instead, the application program must rewrite the record by using the base data set of the VSAM path.

COBOL/400 START Statement

- The START statement does work if the open mode is INPUT.

COBOL/400 DELETE Statement

- Because the sequential file organization must be used to open a VSAM ESDS, it is not possible to delete records in a VSAM ESDS.
- Because the input/output open mode cannot be used to open a VSAM path, it is not possible to delete records in a VSAM path. Instead, the application program must delete the record by using the base data set of the VSAM path. However, if the base data set of the VSAM path is a VSAM ESDS, the DELETE does not work.

C/400 Considerations

The following sections summarize considerations for using C/400 programming language to access remote VSAM data sets from an AS/400 system.

C/400 Open Considerations: Because C/400 programming language supports only sequential I/O, opens will fail if KSDS or VSAM paths are opened.

Open Mode Considerations: Figure E-6 on page E-7 shows the open mode considerations when using C/400 programming language.

Figure E-6. Using C/400 Programming Language to Open a CICS File

C/400 Open Mode	VSAM Data Set Organization			
	ESDS	RRDS	KSDS	VSAM Path
r, rb	X	X	X	X
w, wb	E	E	E	E
w+, wb+, w+b, a+, ab+, a+b, r+, rb+, r+b, a, ab	X	X	X	E
a, ab	X	—	—	—

Where:

X Open mode is allowed.

E Open mode is allowed by C/400 programming language, but the open fails when accessing the VSAM data set.

— Open mode is not applicable.

RPG/400 Considerations

The following sections summarize the limitations that exist when using AS/400 System/36-Compatible RPG II or RPG/400 programming language on an AS/400 system to access remote VSAM data sets. These limitations are in addition to those already stated in this manual.

File Description Specifications: AS/400 users can access remote VSAM data sets with AS/400 System/36-Compatible RPG II or RPG/400 programming language. However, not all RPG/400 processing methods selected by the file description specifications can access remote VSAM data sets. Refer to the following tables to determine which file description specifications to use:

- Figure E-7 on page E-9 for accessing VSAM ESDS
- Figure E-8 on page E-9 for accessing VSAM RRDS
- Figure E-9 on page E-9 for accessing VSAM KSDS
- Figure E-10 on page E-10 for accessing VSAM paths

If a file description specification other than what appears in these tables is used, CICS/DDM rejects the request for opening the file.

Figure E-7. RPG/400 Processing Methods for Remote VSAM ESDS

Type of Processing	Column Number						
	15	16	19	28	31	32	66
Consecutive	I I I I U U U	P S T D P S D	F F F F F F F				
Add Records Only	O						A

Figure E-8. RPG/400 Processing Methods for Remote VSAM RRDS

Type of Processing	Column Number						
	15	16	19	28	31	32	66
Consecutive	I I I U U U	P S D P S D	F F F F F F				
Random by Chain See note. See note.	 I I U U	 C C C C	 F F F F	 R R R R			 A A
Random by Addrout	I I I U U U	P S F P S F	F F F F F F	R R R R	I I I I		
Consecutive and/or Random See note. See note.	 I I U U	 F F F F	 F F F F				 A A

Note: A **K** must be used in column 53 and **RECNO** must be in columns 54 through 59 to indicate relative record number processing.

Figure E-9. RPG/400 Processing Methods for VSAM KSDS

Type of Processing	Column Number						
	15	16	19	28	31	32	66
Sequential							
By key, no add	I	P	F			I	
By key, no add	I	S	F			I	
By key, no add	I	D	F			I	
By key, with add	I	P	F			I	A
By key, with add	I	S	F			I	A
By key, with add	I	D	F			I	A
By key, no add	U	P	F			I	
By key, no add	U	S	F			I	
By key, no add	U	D	F			I	
By key, with add	U	P	F			I	A
By key, with add	U	S	F			I	A
By key, with add	U	D	F			I	A
By limits	I	P	F	L		I	
By limits	I	S	F	L		I	
By limits	I	F	F	L		I	
By limits	U	D	F	L		I	
By limits	U	P	F	L		I	
By limits	U	D	F	L		I	
By limits	U	F	F	L		I	
By limits, adds	I	F	F	L		I	A
By limits, adds	I	F	F	L		I	A
Random by Chain							
No adds	I	C	F	R		I	
With adds	I	C	F	R		I	A
No adds	U	C	F	R		I	
With adds	U	C	F	R		I	A
Random by Addrout	I	P	F	R	I	I	
	I	S	F	R	I	I	
	U	P	F	R	I	I	
	U	S	F	R	I	I	
Sequential and/or Random							
By key, no add	I	F	F			I	
By key, with add	I	F	F			I	A
By key, no add	U	F	F			I	
By key, no add	U	F	F			I	A
Add Records Only	O					I	A

Figure E-10. Processing Methods for Remote VSAM Paths

Type of Processing	Column Number						
	15	16	19	28	31	32	66
Sequential							
By key, no add	I	P	F			I	
By key, no add	I	S	F			I	
By key, no add	I	D	F			I	
By limits	I	P	F	L		I	
By limits	I	S	F	L		I	
By limits	I	F	F	L		I	
By limits	I	D	F	L		I	
Random by Chain							
No adds	I	C	F	R		I	
Random by Addrout	I	P	F	R	I	I	
	I	S	F	R	I	I	
Sequential and/or Random							
By key, no add	I	F	F			I	

RPG/400 Input/Output Operations: Be aware of the following when accessing remote VSAM data sets:

- Records can be read or added by relative record number only when the remote VSAM data set is an RRDS. Random processing by relative record number can be used only when processing a VSAM RRDS.
- A request to delete a record in an ESDS does not work because ESDS is never delete-capable.
- The processing method cannot use the update or output specification in column 15 when the remote VSAM data set is a VSAM path. Instead, all add, update, or delete record requests must be made via the base data set of the VSAM path. However, if the base data set of the VSAM path is a VSAM ESDS, the DELETE does not work.
- The READP operation code cannot be used to read the previous record in a VSAM path.
- Because VSAM KSDS does not allow duplicate keys, a request to add a record that duplicates the key of an existing record in a KSDS does not work.
- When accessing a KSDS, an update request that changes the key value of the record does not work.
- For RPG/400 programming language, *HIVAL can be used to obtain the last record of a remote KSDS. However, *HIVAL does not work when accessing a VSAM path.

Appendix F. DDM Differences

This appendix describes the DDM differences between:

- AS/400 system and System/36
- AS/400 system and System/38

For additional information on the differences between the AS/400 system and the System/36 or System/38 see *Concepts and Programmer's Guide for the System/36 Environment* or *System/38 Environment Programmer's Guide/Reference*.

AS/400 System and System/36 DDM Differences

The following is a list of differences between the AS/400 system and System/36:

- The network resource directory (NRD) procedures are not supported on the AS/400 system.
  - The System/36 NRD used one or more libraries containing DDM files on the AS/400 system. One System/36 NRD entry is equivalent to one DDM file on the AS/400 system.
    - Use the Work with DDM Files (WRKDDMF) command in place of the EDITNRD and DELNRD procedures.
    - Use the Display DDM Files (DSPDDMF) command in place of the LISTNRD procedure.
    - Use the Restore Object (RSTOBJ) command in place of the RESTNRD procedure.
    - Use the Save Object (SAVOBJ) command in place of the SAVNRD procedure.

    See the *System/36 Migration Aid User's Guide and Reference* for additional information.
  - If an NRD entry on the System/36 refers to a remote file, and a SAVE or RESTORE procedure is requested, the System/36 saves or restores the data of the remote file. The AS/400 Save Object (SAVOBJ) or Restore Object (RSTOBJ) command saves or restores only the definition of the DDM file, not the remote data.
  - When using date-differentiated files on the System/36, the most current file is selected. When running from a System/36 to an AS/400 system, the NRD entry must specify a member name of *LAST to access the last member of the database file, which is the file with the most recent date. If no member name is specified, *FIRST is used.
- The remote label in the NRD on a System/36 source must be in the syntax required by the target system. For further information on specific naming conventions, see "Specifying Target System File Names" on page 6-2.
- The number of records allocated for a file differs between the System/36 and the AS/400 system. File space allocation on the System/36 is in block units (2560 bytes) while on the AS/400 system, file space allocation is by number of records. For example, if a user asks for a sequential file capable of storing ten 100-byte records, the System/36 allocates 1 block of file space (2560 bytes), and uses as much of the file space as possible (2500 bytes), giving the user twenty-five 100-byte records.

  The AS/400 system allocates exactly 10 records. If the user took advantage of this allocation method on the System/36, the file (nonextendable) created using DDM on the AS/400 system might be too small.
- The DDM Change End-of-File (CHGEOF) command used on the System/36 is not supported on the AS/400 system. (FORTRAN/400 is the only language affected.)
- The AS/400 system does not support writing over data on the Delete File (DLTF) command. If an AS/400 user accessing a System/36 wants to overwrite the data, an application program must be written on the AS/400 system, or the user must access the target System/36 and perform the delete operation with the erase operation.

- The System/36 does not allow a record with hex FF in the first byte to be inserted into a delete-capable file. The AS/400 system allows this.
- When running System/36 applications to another System/36, the application waits indefinitely for the resource to become available. When running System/36 applications to or from an AS/400 system, these applications may result in time-outs while waiting for a resource to become available.
- Direct files are extendable on the System/36 but not on the AS/400 system. If a direct file is created on the AS/400 system as extendable, the file is allocated with three extents, but is never extended beyond the initial size plus three extents.
- The System/36 relay function is *not* supported whenever one of the systems in the network along the path from the source system to the target system is not a System/36. The AS/400 system *never* supports the relay function; advanced peer-to-peer networking (APPN) must be used.
- Key fields on the System/36 are considered to be strictly character fields. The System/36 does not recognize a key field as being packed or zoned. Unexpected results may occur if source AS/400 application programs refer to packed fields within a System/36 file. Because of the way packed numbers are stored and the fact that the System/36 does not recognize key fields as packed on relative keyed operations, records may be returned in a different order than expected or no record may be found when one is expected.

  As an example, the RPG/400 SETLL statement may fail unexpectedly with record not found or may select a record other than the record expected when using packed fields to a System/36 file. Only character and unsigned numeric fields should be used as key fields.

AS/400 System and System/38 DDM Differences

The following is a list of differences between the AS/400 system and System/38:

- Three parameters are added to the Create DDM File (CRTDDMF) and Change DDM File (CHGDDMF) commands. These parameters are the remote location name (RMTLOCNAME), local location name (LCLLOCNAME), and the remote network ID (RMTNETID). DDM files can be created either in the System/38 environment or on the AS/400 system.
- The Submit Remote Command (SBMRMTCMD) command must be in the syntax of the target system, even in the System/38 environment. For example, a System/38 submitting commands to an AS/400 system must use the syntax of the AS/400 system.
- The remote file name must be in the syntax of the target system. For further information on specific naming conventions, see "Specifying Target System File Names" on page 6-2.
- The default value for the DDMACC parameter on the Change Network Attributes (CHGNETA) command on the System/38 is *REJECT. The default value for the DDMACC parameter on the AS/400 system is *OBJAUT.
- On the System/38, files are created as FIFO (first in, first out) or LIFO (last in, first out). The default for creating a file is FIFO on the System/38.

  When running System/38 applications that are dependent on duplicate keys being FIFO or LIFO to an AS/400 system, you should specify either FIFO or LIFO when creating your AS/400 files because there is no default for AS/400 files. This means the AS/400 system looks for an available index path to share, which could be either FIFO or LIFO.
- Keyed files containing fields other than character (zoned or packed) created via DDM on a remote System/38 may result in the fields defined as character fields. This may produce unexpected results when such a file is processed using relative keyed operations. Because the file is created with fields that are not packed, records may be returned in a different order than expected or no record may be found when one is expected.

  As an example, the RPG/400 SETLL statement may fail unexpectedly with record not found or may select a record other than the

record expected when using packed fields to a System/38 file. Only character and unsigned numeric fields should be used as key fields for files that are created via DDM on the remote System/38.

- To support adding a record by the relative record number operation, an RPG/400 program is required for DDM to do a READ CHAIN(RRN) operation followed by a WRITE operation. The file must be opened for read and update authorities, and the user must have read and update data authorities to the file.

  Format selector programs on adding a record by the relative record number operation are only supported on the AS/400 system. Incompatibilities may arise for those users who have a format selector program for a logical file if they do direct file processing.

Bibliography

The manuals listed in this bibliography are suggested for finding more information about subjects in this publication. Not all of these manuals are referred to in this guide. You may need to use one or more of the following IBM AS/400 manuals while using this guide. The manuals are listed with their full title and base order number. When these manuals are referred to in this guide, a shortened version of the title listed is used.

Communications:

- *Communications: Advanced Peer-to-Peer Networking Guide*, SC41-8188, provides the application programmer with information about the advanced peer-to-peer networking (APPN) support provided by the AS/400 system. This guide provides information for configuring an APPN network and presents considerations to apply when using APPN.

  Short Title: *APPN Guide.*

- *Communications: Advanced Program-to-Program Communications Programmer's Guide*, SC41-8189, provides the application programmer with information about the advanced program-to-program communications (APPC) support provided by the AS/400 system. This is a guide for programming and defining the communications environment for APPC communications.

  Short Title: *APPC Programmer's Guide.*

- *Communications: Distribution Services Network Guide*, SC41-9588, provides the system operator or administrator with information about configuring a network for Systems Network Architecture distribution services (SNADS) and the Virtual Machine/Multiple Virtual Storage (VM/MVS) bridge.

  Short Title: *Distribution Services Network Guide.*

- *Communications: Intersystem Communications Function Programmer's Guide*, SC41-9590, provides the application programmer with information needed to write application programs that use AS/400 communications and the OS/400 intersystem communications function (OS/400-ICF).

  Short Title: *ICF Programmer's Guide.*

- *Communications: Management Guide*, SC41-0024, provides the system operator with communications work management information, error handling information, communications status information, and communications performance information.

  Short Title: *Communications Management Guide*

- *Communications: Operating System/400* Communications Configuration Reference*, SC41-0001, provides the application programmer with information on configuring line, controller, and device descriptions to communicate within a network. Additional configuration considerations are discussed.

  Short Title: *OS/400* Communications Configuration Reference.*

- *Communications: Remote Work Station Guide*, SC41-0002, provides the system administrator or end user with concepts, examples, and information on preparation and configuration for using the display station pass-through function. This guide also contains information about using 3270 remote attachment, the Distributed Host Command Facility (DHCF) network, and the X.21 short hold mode (SHM) network.

  Short Title: *Remote Work Station Guide.*

- *Communications: X.25 Network Guide*, SC41-0005, provides the system administrator or end user with information about setting up, configuring, and using AS/400 systems through an X.25 network.

  Short Title: *X.25 Network Guide.*

- *Distributed Relational Database Guide*, SC41-0025, provides database administrators and system programmers with information for planning, setting up, programming, administering, and operating a distributed relational database on an AS/400 system.

  Short Title: *Distributed Database Guide.*

Languages:

- *Languages: BASIC User's Guide and Reference*, SC09-1157
- *Languages: PL/I Reference Summary*, SX09-1051
- *Languages: PL/I User's Guide and Reference*, SC09-1156
- *Languages: Systems Application Architecture* AD/Cycle* RPG/400* User's Guide*, SC09-1348
- *Languages: Systems Application Architecture* C/400* User's Guide*, SC09-1347
- *Languages: System/36-Compatible COBOL User's Guide and Reference*, SC09-1160
- *Languages: System/36-Compatible RPG II User's Guide and Reference*, SC09-1162
- *Languages: System/38-Compatible COBOL User's Guide and Reference*, SC09-1159

Planning and Installation:

- *System/36 Migration Aid User's Guide and Reference*, SC09-1106

Programming:

- *Data Management Guide*, SC41-9658, provides the application programmer with information about using files in application programs. This manual includes information on fundamental structure and concepts of data management support on the system, data management support for display stations, tapes, diskettes, and spooling support.

  Short Title: *Data Management Guide.*

- *Programming: Concepts and Programmer's Guide for the System/36 Environment*, SC41-9663, provides the application programmer with information identifying the differences in the applications process in the System/36 environment on the AS/400 system. This includes an environment functional overview, considerations for migration, programming, communications, security, and coexistence.

  Short Title: *Concepts and Programmer's Guide for the System/36 Environment.*

- *Programming: Control Language Reference*, SC41-0030, provides the application programmer with a description of the AS/400 control language (CL) and its commands.

  Short Title: *CL Reference.*

- *Data Description Specifications Reference*, SC41-9620, provides the application programmer with detailed descriptions of the entries and keywords needed to describe database files (both logical and physical) and certain device files (for displays, printers, and intersystem communications function)) external to the user's program.

  Short Title: *DDS Reference.*

- *Database Guide*, SC41-9659, provides the application programmer with a detailed discussion of the AS/400 database organization, including information on how to create, describe, and update database files on the system.

  Short Title: *Database Guide.*

- *Programming: System/38 Environment Programmer's Guide and Reference*, SC41-9755, provides the application programmer with the information needed to migrate from a System/38, convert to an AS/400 system, and coexist in a network.

  Short Title: *System/38 Environment Programmer's Guide/Reference.*

Utilities:

- *Application Development Tools: Data File Utility User's Guide and Reference*, SC09-1381
- *Communications: Remote Job Entry User's Guide and Reference*, SC09-1168
- *Utilities: Sort User's Guide and Reference*, SC09-1363

PC Support/400:

- *PC Support/400: DOS and OS/2 Technical Reference*, SC41-8091, provides the programmer or personal computer technical coordinator with technical information needed to do advanced configuration or tailoring of PC Support/400 using DOS and the OS/2 program.

  Short Title: *PC Support/400 Technical Reference for DOS and OS/2.*

- *PC Support/400: DOS User's Guide*, SC41-8199, provides personal computer users with a personal computer attached to an AS/400 system with information about operating and using PC Support/400 using DOS.

  Short Title: *PC Support/400 User's Guide for DOS.*

- *PC Support/400: OS/2 User's Guide*, SC41-8200, provides personal computer users with a personal computer attached to an AS/400 system with information about operating and using PC Support/400 using the OS/2 program.

  Short Title: *PC Support/400 User's Guide for OS/2.*

Distributed Data Management (DDM) Architecture:

- *Distributed Data Management Architecture: General Information*, GC21-9527
- *Distributed Data Management Architecture: Implementation Planner's Guide*, GC21-9528
- *Distributed Data Management Architecture: Implementation Programmer's Guide*, SC21-9529
- *Distributed Data Management Architecture: Reference*, SC21-9526

You may need to use the following IBM manuals when using DDM on systems other than an AS/400 system:

- *CICS/OS/VS Resource Definition (Macro)*, SC33-0149
- *Customer Information Control System/Distributed Data Management User's Guide*, SC21-8066
- *Distributed Data Management Technical Reference for the IBM Personal Computer*, SC21-9644
- *System/36 Distributed Data Management Guide*, SC21-8011
- *System/38 Distributed Data Management Guide*, SC21-8036
- *Using Distributed Data Management for the IBM Personal Computer*, SC21-9643

Index

C

D

E

F

O

P

Q

R

S

T

U

W

Special Characters

Reader's Comments

Application System/400
Distributed Data
Management Guide
Version 2

Publication No. SC41-9600-00

Use this form to tell us what you think about this manual. If you have found errors in it, or if you want to express your opinion about it (such as organization, subject matter, appearance) or make suggestions for improvement, this is the form to use.

To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer. This form is provided for comments about the information in this manual and the way it is presented.

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.

Be sure to print your name and address below if you would like a reply.

Name

Address

Company or Organization

Phone No.

Reader's Comments
SC41-9600-00

IBM®

Cut or
Along

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

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN DEPT 245
IBM CORPORATION
3605 HWY 52 N
ROCHESTER MN 55901-7899

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

Cut or
Along

Program Number: 5738-SS1

Printed in Denmark by From & Co.

SC41-9600-00