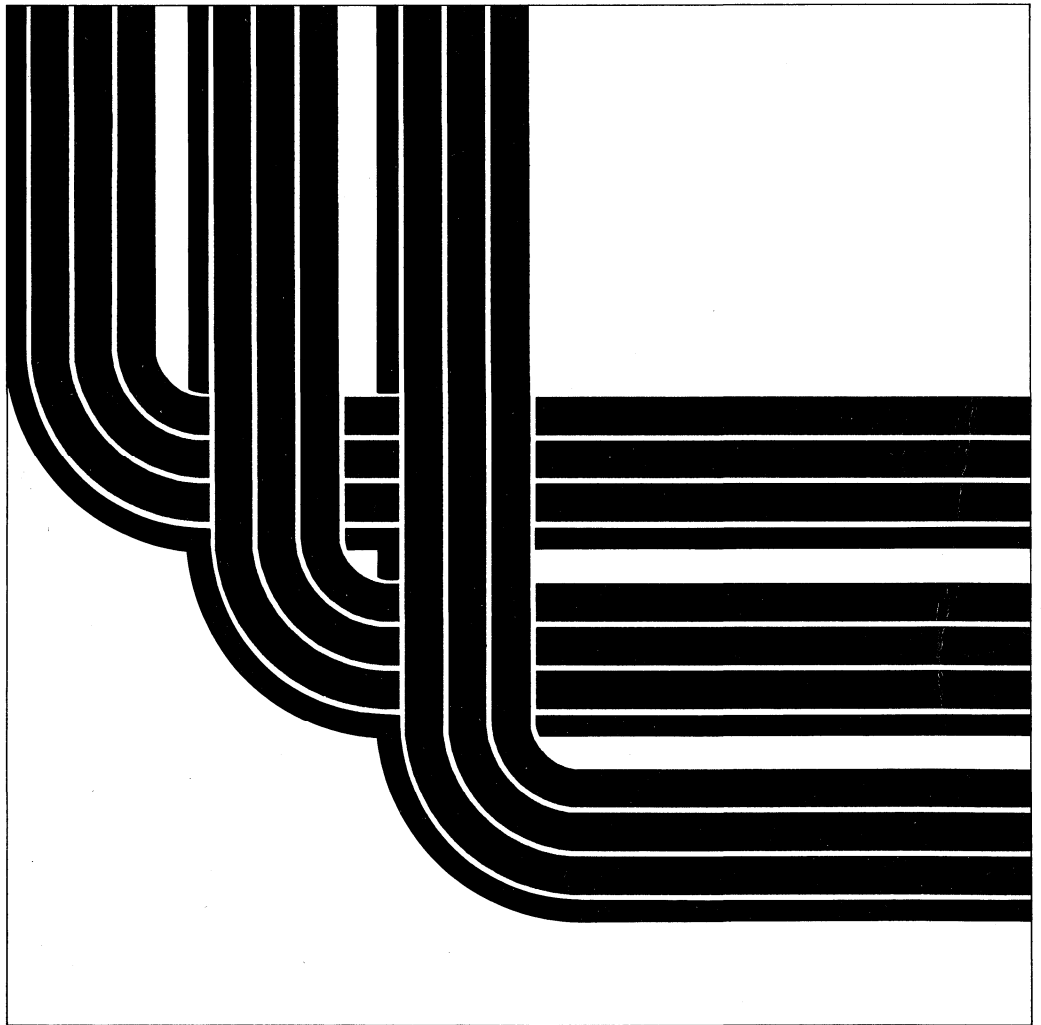


**Communications:
Intersystem Communications Function
Programmer's Guide**

Version 2



Take Note!

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

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.

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.

© Copyright International Business Machines Corporation 1991. All rights reserved.

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	xi	Starting Communications with the Remote System	3-5
Programming Interface	xi	Starting a Program on the Remote System	3-7
About This Guide	xiii	Connecting to the Session — Target Program	3-8
Who Should Use This Guide	xiii	Sending and Receiving Data	3-11
Chapter 1. Introduction to AS/400 System Communications	1-1	Ending Communications with the Remote System	3-12
Planning for Data Communications	1-1	Ending the Transaction	3-12
Installing Communications Hardware	1-1	Ending the Session	3-13
Configuring Your System for Data Communications	1-1	Closing the Intersystem Communications Function File	3-14
Writing Programs that Use the Intersystem Communications Function (ICF)	1-1	Varying off Communications Configurations	3-14
Operating Communications on the AS/400 System	1-1	Additional Information on Sessions and Transactions	3-14
Chapter 2. Communications Features	2-1	Multiple Transactions	3-14
Intersystem Communications Function Communications Types	2-1	Multiple Sessions	3-16
AS/400 System Communications Types	2-1	Summary	3-17
Non-Intersystem Communications Function Communications	2-3	Source Program	3-17
Communicating with Remote Work Stations	2-4	Target Program	3-19
Combinations of Communications Types	2-4	Chapter 4. Intersystem Communications Function Files	4-1
AS/400 System Communications Line Support	2-4	Introduction to Intersystem Communications Function Files	4-1
Operating System/400 Communications Configuration	2-4	Intersystem Communications Function File Commands	4-2
Intersystem Communications Function File	2-5	File-Level Attribute Commands	4-2
Data Description Specifications (DDS)	2-5	Program Device Entry Commands	4-3
System-Supplied Formats	2-5	Display Information Commands	4-3
Control Language	2-5	Creating an Intersystem Communications Function File	4-3
Security	2-5	Defining the Record Formats for an Intersystem Communications Function File	4-3
Error Handling	2-5	File Attributes	4-4
High-Level Language Support	2-6	Acquiring a Program Device when the File Is Opened	4-4
Additional Programming Support	2-6	Changing an Intersystem Communications Function File	4-7
Chapter 3. Introduction to Intersystem Communications Function	3-1	Overriding an Intersystem Communications Function File	4-7
Configuring for Communications	3-2	Identifying the Devices Used with an Intersystem Communications Function File	4-8
Varying on Communications Configurations	3-3	Defining Program Device Entries	4-9
The Intersystem Communications Function File	3-3	Mapping Program Device Name to Communications Configurations	4-10
Defining the File	3-3	Communications-Type-Dependent Attributes	4-14
Using the File	3-4	Intersystem Communications Function Command Summary	4-17
Starting Your Program	3-4		
Opening the Intersystem Communications Function File	3-5		

Chapter 5. Using an Intersystem Communications Function File	5-1	Record-Identification (RECID)	6-8
Opening an Intersystem Communications Function File	5-1	Problem Notification	6-9
Obtaining Information about the Open Intersystem Communications Function File	5-2	Fail (FAIL)	6-9
Acquiring a Program Device	5-2	Cancel (CANCEL)	6-10
Acquiring a Program Device — Source Program	5-3	Negative-Response (NEGRSP)	6-11
Acquiring a Program Device — Target Program	5-3	Additional Keywords	6-12
Obtaining Information about a Particular Program Device	5-4	Respond-to-Confirm (RSPCONFIRM)	6-12
Program Device Definition List	5-4	Request-to-Write (RQSWRT)	6-13
Get-Attributes Operation	5-4	Allow-Write (ALWWRT)	6-15
Sending and Receiving Data	5-6	Cancel-Invite (CNLINVITE)	6-16
Common I/O Feedback Area	5-6	Ending a Communications Transaction	6-16
File-Dependent I/O Feedback Area	5-6	Detach (DETACH)	6-16
Checking Return Codes	5-7	Ending the Communications Session	6-17
Writing to a Program Device	5-8	End-of-Session (EOS)	6-17
Inviting a Program Device	5-8	Using Response Indicator Keywords	6-18
Format Selection Processing	5-9	Receive-Confirm	6-18
Reading from Invited Program Devices	5-11	Receive-End-of-Group	6-19
Reading from One Program Device	5-16	Receive-Function-Management-Header	6-19
Writing and Then Reading from One Program Device	5-17	Receive-Fail	6-19
Canceling an Invite of a Program Device	5-17	Receive-Cancel	6-19
Waiting for a Display File, an ICF File, and a Data Queue	5-17	Receive-Negative-Response	6-19
Releasing a Program Device	5-19	Receive-Turnaround	6-19
Closing an Intersystem Communications Function File	5-19	Receive-Detach	6-19
Summary	5-19	Example DDS Files for Creating an Intersystem Communications Function File Keyword Processing Charts	6-19 6-22
Local System	5-20		
Remote System	5-21		
Chapter 6. Using Communications DDS Keywords	6-1	Chapter 7. Using System-Supplied Communications Formats	7-1
Starting a Program on the Remote System	6-1	General Description	7-1
Evoke (EVOKE, SECURITY, and SYNLVL)	6-1	Starting a Program on the Remote System	7-2
Sending Data	6-4	Evoke	7-2
Variable-Length Data (VARLEN)	6-4	Sending Data	7-4
Variable-Buffer-Management (VARBUFMGMT)	6-4	Receiving Data	7-6
Force-Data (FRCDTA)	6-4	Invite	7-6
Confirm (CONFIRM)	6-4	Timer	7-7
Format-Name (FMTNAME)	6-5	Problem Notification	7-8
Subdevice-Selection (SUBDEV)	6-5	Fail	7-8
End-of-Group (ENDGRP)	6-5	Cancel	7-10
Function-Management-Header (FMH)	6-5	Negative-Response	7-12
Examples of Sending Data	6-6	Additional System-Supplied Formats	7-14
Receiving Data	6-7	Positive-Response	7-14
Invite (INVITE)	6-7	Request-to-Write	7-14
Timer (TIMER)	6-8	Cancel-Invite	7-17
		Ending a Communications Transaction	7-18
		Detach	7-18
		Ending the Communications Session	7-20
		End of Session	7-20
		System-Supplied Format Support	7-22
		Mapping System-Supplied Formats to DDS Keywords	7-23
		Chapter 8. Programming Considerations	8-1
		Return Codes	8-1
		Major Codes	8-1

Minor Codes	8-1	Displaying Communications Status	12-7
ICF Environment Considerations	8-1	Appendix A. Language Operations, Data Description Specifications Keywords, and System-Supplied Formats	
Open or Acquire Considerations	8-2	Language Operations	A-1
Output Considerations	8-2	DDS Keyword Support	A-2
Input Considerations	8-3	System-Supplied Format Support	A-6
Release, End-of-Session, and Close Considerations	8-4	Appendix B. Communications Error Handling	
Release Considerations	8-4	System Error Classification	B-1
End-of-Session Considerations	8-4	System Messages	B-1
Close Considerations	8-5	User Program Error Detection	B-2
Remote Program Start Considerations	8-5	Control Language (CL) Commands for Determining Configuration Status	B-3
Defining the Environment	8-5	Major/Minor Return Codes	B-3
Handling Program Start Requests	8-5	Major Code 00	B-4
Prestarting Jobs for Program Start Requests	8-8	Major Code 02	B-6
Commands	8-8	Major Code 03	B-9
Application Considerations	8-9	Major Code 04	B-10
Security Considerations for Prestart Jobs	8-9	Major Codes 08 – 11	B-11
Prestart Jobs Program	8-10	Major Code 34	B-11
System Considerations	8-12	Major Code 80	B-12
Security Considerations	8-12	Major Code 81	B-13
File Considerations	8-13	Major Code 82	B-14
File Redirection	8-13	Major Code 83	B-18
Additional Considerations	8-13	Failed Program Start Requests	B-21
Chapter 9. Communications Applications with C/400 Programming Language			
Introduction to the C/400 Interface	9-1	Appendix C. Open Feedback and I/O Feedback	
Multiple-Session Inquiry	9-2	Open Feedback Area	C-1
Error Handling	9-2	Program Device Definition List	C-2
Accessing the Feedback Areas	9-2	Input/Output Feedback Area	C-3
Source Program Multiple-Session Inquiry	9-10	Common I/O Feedback Area	C-4
File-Dependent I/O Feedback Area C-5			
Appendix D. EBCDIC and ASCII Character Sets			
EBCDIC Character Set D-1			
ASCII Character Set D-2			
Appendix E. File Transfer Support			
File Transfer Support Overview E-1			
File Transfer Considerations E-1			
To and From an AS/400 System E-2			
AS/400 System Retrieving from System/36 E-2			
AS/400 System Sending to System/36 E-2			
Multiple Communication-Type Support E-2			
File Transfer Parameters E-4			
To and From an AS/400 System E-4			
AS/400 System Sending to System/36 E-6			
AS/400 System Retrieving a File from System/36 E-9			
Chapter 10. Communications Applications with COBOL/400 Programming Language			
Introduction to the COBOL/400 Interface	10-1		
Example Programs	10-2		
Batch Data Transfer (Example I)	10-4		
Multiple-Session Inquiry (Example II)	10-25		
Chapter 11. Communications Applications with RPG/400 Programming Language			
Introduction to the RPG/400 Interface	11-1		
Example Programs	11-3		
Batch Data Transfer (Example I)	11-4		
Multiple-Session Inquiry (Example II)	11-26		
Chapter 12. Tracing Intersystem Communications Function Operations and Functions			
Starting the Trace	12-1		
Stopping the Trace	12-2		
Trace Records Sent to a Spooled File	12-3		
Trace Records Sent to a Database File	12-5		
Ending the Trace	12-6		
Additional Considerations	12-6		

Retrieving a Library Member from System/36	E-11
Calling File Transfer Support for C/400 Programming Language	E-14
Calling File Transfer Support for COBOL/400 Programming Language	E-21
Calling File Transfer Support for RPG/400 Programming Language	E-25
Calling File Transfer Support for a CL Program	E-28

File Transfer Support Messages	E-29
--	------

Bibliography	H-1
Communications Manuals	H-1
Programming Language Manuals	H-2
System/36-Related Manuals	H-3

Index	X-1
------------------------	-----

Figures

3-1.	Sending Data from a Local Program to a Remote Program	3-1	5-3.	Using Write Operation When Sending Data	5-8
3-2.	Major Parts of ICF Data Management	3-2	5-4.	Format Selection Options	5-11
3-3.	ICF File-Configuration Relationship	3-4	5-5.	Using the Invite Function and Read-from-Invited-Program-Devices Operation to Receive Data	5-12
3-4.	Establishing a Session	3-5	5-6.	Relationship between Timer and Read-from-Invited-Program-Devices Operations	5-14
3-5.	Communications Session Established	3-6	5-7.	Using the Read Operation	5-17
3-6.	Program Started at Remote System by Evoke Function	3-7	5-8.	Display File and ICF File Entry Field Attributes	5-18
3-7.	Requesting Program Device Relationship	3-9	5-9.	Relationship of Program, File, and Configuration	5-20
3-8.	Establishing a Logical Connection between the Target Program and the Session	3-10	5-10.	C/400 Program, File, and Configuration Mapping	5-22
3-9.	Data Sent by a Send Request	3-11	5-11.	COBOL/400 Program, File, and Configuration Mapping	5-23
3-10.	Ending a Communications Transaction: Detach Function	3-12	5-12.	RPG/400 Program, File, and Configuration Mapping	5-24
3-11.	Ending a Session: Release Operation and End-of-Session Function	3-13	6-1.	Starting a Target Program	6-3
3-12.	Starting and Ending Sessions and Transactions	3-15	6-2.	Using the CONFIRM, FRCDDTA, and FMTNAME Keywords to Send Data	6-6
3-13.	Remotely Started Program Starts a Session and Transaction	3-16	6-3.	Using the ENDGRP and FMH Keywords to Send Data	6-7
3-14.	The AS/400 System Application Starts a Session with a Remote System	3-18	6-4.	Using the FAIL Keyword to Send an Error Indication	6-10
3-15.	Remote System Starts a Session with a Program Start Request	3-19	6-5.	Using the CANCEL Keyword to Send an Error Indication	6-11
4-1.	ICF File Overview	4-2	6-6.	Sending a Negative Response with Sense Code to Remote System	6-12
4-2.	Creating an ICF File	4-3	6-7.	Using the Respond-to-Confirm Function	6-13
4-3.	ICF File Attributes	4-4	6-8.	Using the Request-to-Write Function	6-14
4-4.	File Attributes for Changing an ICF File	4-7	6-9.	Using the Allow-Write Function	6-15
4-5.	File Attributes for Overriding an ICF File	4-7	6-10.	Using the Cancel-Invite Function	6-16
4-6.	Defining a Program Device Entry to an ICF File	4-9	6-11.	Ending a Communications Transaction	6-17
4-7.	Mapping Parameters for All Communications Types	4-10	6-12.	Using the Release and End-of-Session Functions	6-18
4-8.	Relationship of Remote Location Name to Device Description	4-13	6-13.	DDS Source File for a Batch Data Transfer Program	6-20
4-9.	Communications-Dependent Attributes	4-14	6-14.	DDS Source File for a Multiple Session Program	6-21
4-10.	Relationship between ICF Commands	4-18	6-15.	Output DDS Processing Keyword Support	6-22
5-1.	Relationship of Program Device Entries to Operations	5-3	6-16.	Input DDS Processing Keyword Support	6-23
5-2.	Attribute Information Fields	5-4	6-17.	Keyword Processing Chart	6-24
			7-1.	Starting a Target Program	7-2

7-2.	Invoke Parameter List	7-3	9-10.	Target Program Example — CTGTDMUL (User-Defined Formats)	9-25
7-3.	Invoke RPG/400 Output Specification	7-4	10-1.	COBOL/400 Statements	10-1
7-4.	Using \$\$SENDNF, \$\$SENDNI, and \$\$SENDE to Send Data	7-5	10-2.	File Status Values for Major and Minor Return Codes	10-2
7-5.	Required Output Fields	7-5	10-3.	Batch Data Transfer	10-3
7-6.	Send RPG/400 Output Specification	7-6	10-4.	Multiple-Session Inquiry	10-3
7-7.	Timer RPG/400 Output Specification	7-8	10-5.	Evoke Request Starts a Target Program	10-4
7-8.	Using \$\$FAIL to Send an Error Signal	7-9	10-6.	Target Program Prints Records	10-4
7-9.	Fail RPG/400 Output Specification	7-10	10-7.	Source Program Prints the Received Records	10-5
7-10.	Using \$\$CANL to Send an Error Indication	7-11	10-8.	Target Program Ends the Transaction	10-5
7-11.	Cancel RPG/400 Output Specification	7-12	10-9.	Source Program Example — CSDBAT (User-Defined Formats)	10-8
7-12.	Using \$\$NRSP to Send an Error Condition	7-13	10-10.	Source Program Example — CSFBAT (System-Supplied Formats)	10-13
7-13.	Sense Data Format	7-13	10-11.	Target Program Example — CTDBAT (User-Defined Formats)	10-19
7-14.	Negative-Response RPG/400 Output Specifications	7-14	10-12.	Target Program Example — CTFBAT (System-Supplied Formats)	10-23
7-15.	Using \$\$RCD to Request Write	7-15	10-13.	Program Starts at Display Station	10-26
7-16.	Request-to-Write RPG/400 Output Specification	7-16	10-14.	Program Devices Explicitly Acquired	10-27
7-17.	Using \$\$CNLINV to Cancel an Invite	7-17	10-15.	Evoke Starts Target Programs	10-28
7-18.	Cancel-Invite RPG/400 Output Specifications	7-18	10-16.	Main Menu Written to Display Station	10-29
7-19.	Ending the Communications Transaction	7-19	10-17.	Program Sends Inquiry Request to Remote System	10-30
7-20.	Detach RPG/400 Output Specification	7-20	10-18.	Target Program Sends a Reply	10-31
7-21.	Using the Release Operation and End-of-Session Function	7-21	10-19.	Program Ends the Session	10-32
7-22.	End-of-Session RPG/400 Output Specification	7-22	10-20.	DDS for Source Program Multiple Session Inquiry Using CMNFIL	10-33
7-23.	System-Supplied Format Support	7-22	10-21.	DDS for Source Program Multiple Session Inquiry Using DSPFIL	10-34
7-24.	Mapping of System-Supplied Formats to DDS Keywords	7-23	10-22.	Source Program Example — CSDMUL (User-Defined Formats)	10-38
8-1.	Sample ICF Communications Environment	8-7	10-23.	Source Program Example — CSFMUL (System-Supplied Formats)	10-50
8-2.	COBOL/400 Coding for a Prestart Job Program	8-10	10-24.	DDS Source for ICF File Used in Target Program Multiple Session Inquiry	10-61
9-1.	C/400 Function	9-1	10-25.	DDS Source for Database File Used in Target Program Multiple Session Inquiry	10-61
9-2.	Program Starts at Display Station	9-3	10-26.	Target Program Example — CTDMUL (User-Defined Formats)	10-63
9-3.	Program Devices Explicitly Acquired	9-4	10-27.	Target Program Example — CTFMUL (System-Supplied Formats)	10-68
9-4.	Evoke Starts Target Programs	9-5			
9-5.	Main Menu Written to Display Station	9-6			
9-6.	Program Sends Inquiry Request to Remote System	9-7			
9-7.	Target Program Sends a Reply	9-8			
9-8.	Program Ends the Session	9-9			
9-9.	Source Program Example — CSRCDMUL (User-Defined Formats)	9-13			

11-1.	RPG/400 Statements	11-1	12-1.	Spooled Trace Records	12-4
11-2.	*STATUS Values for Major and Minor Return Codes	11-2	A-1.	Language Operations	A-1
11-3.	Batch Data Transfer	11-3	A-2.	Language Operations	A-2
11-4.	Multiple-Session Inquiry	11-4	A-3.	Processing Control DDS Keywords	A-2
11-5.	Evoke Request Starts a Target Program	11-5	A-4.	System-Supplied Format Support	A-6
11-6.	Target Program Prints Records .	11-5	B-1.	File Error Message Identifier Groups	B-2
11-7.	Source Program Prints the Received Records	11-5	B-2.	Major Code 00	B-4
11-8.	Target Program Ends the Transaction	11-6	B-3.	Major Code 02	B-7
11-9.	Source Program Example — RSDBAT (User-Defined Formats) .	11-8	B-4.	Major Code 03	B-9
11-10.	Source Program Example — RSFBAT (System-Supplied Formats)	11-13	B-5.	Major Code 04	B-10
11-11.	Target Program Example — RTDBAT (User-Defined Formats)	11-19	B-6.	Major Codes 08-11	B-11
11-12.	Target Program Example — RTFBAT (System-Supplied Formats)	11-23	B-7.	Major Code 34	B-11
11-13.	Program Starts at Display Station	11-27	B-8.	Major Code 80	B-12
11-14.	Program Devices Explicitly Acquired	11-28	B-9.	Major Code 81	B-13
11-15.	Evoke Starts Target Programs .	11-29	B-10.	Major Code 82	B-15
11-16.	Main Menu Written to Display Station	11-30	B-11.	Major Code 83	B-18
11-17.	Program Sends Inquiry Request to Remote System	11-31	B-12.	Reason Codes for Rejected Program Start Requests	B-22
11-18.	Target Program Sends a Reply .	11-32	C-1.	Offset Values for COBOL/400, FORTRAN/400, and RPG/400	C-1
11-19.	Program Ends the Session	11-33	C-2.	Open Feedback Area	C-1
11-20.	Source Program Example — RSDMUL (User-Defined Formats)	11-38	C-3.	Program Device Definition List . . .	C-2
11-21.	Source Program Example — RSFMUL (System-Supplied Formats)	11-51	C-4.	Common I/O Feedback Area	C-4
11-22.	DDS Source for ICF File Used by Target Program Multiple-Session Inquiry	11-63	C-5.	File-dependent I/O Feedback Area	C-5
11-23.	DDS Source for Database File Used by Target Program Multiple Session Inquiry	11-63	D-1.	EBCDIC Character Set	D-1
11-24.	Target Program Example — RTDMUL (User-Defined Formats)	11-65	D-2.	ASCII Character Set	D-2
11-25.	Target Program Example — RTFMUL (System-Supplied Formats)	11-69	E-1.	Example of File Transfer Support	E-1
			E-2.	Transferring Files to and from an AS/400 System	E-4
			E-3.	AS/400 System Sending to System/36	E-6
			E-4.	AS/400 System Retrieving a File from System/36	E-9
			E-5.	Retrieving a Library Member from System/36	E-11
			E-6.	C/400 Coding for File Transfer Support	E-14
			E-7.	COBOL/400 Coding for File Transfer Support	E-21
			E-8.	RPG/400 Coding for File Transfer Support	E-25
			E-9.	CL Coding for File Transfer Support	E-28
			E-10.	File Transfer Messages	E-29

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
Operating System/400	OS/400	RPG/400
SAA	Systems Application Architecture	System/370
System/390	VTAM	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 programmer's guide is intended to help application programmers write communications programs that use the intersystem communications function (ICF). It primarily contains reference information which allows the customer to write programs that use the services of ICF. ICF contains no programming interfaces for customers.

About This Guide

This reference guide contains programming information for writing application programs that use the intersystem communications function (ICF).

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 primarily for AS/400 system and remote system application programmers who write communications programs that use ICF.

To work with the information in this guide, you should have knowledge of general communications concepts. AS/400 communications concepts are covered in the manual *System Concepts*, GC41-9802. In addition, specific communications topics are discussed in the online index search. For more information on basic communications, you can also refer to the Discover/ IBM AS/400 course in the communications module. The Discover/ IBM AS/400 course is separately orderable.

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

- AS/400 system programming terminology and programming using the C/400, COBOL/400, FORTRAN/400, or RPG/400 languages.
- Concepts of data communications as described in the manual *Data Communications Concepts*, GC21-5169.

Chapter 1. Introduction to AS/400 System Communications

This chapter describes, in general, the different sources and background information needed to use intersystem communications function (ICF) communications on the IBM® Application System/400® (AS/400®) system. **ICF** is a function of the operating system that allows a program to communicate interactively with another program or system. Detailed instructions are available in other manuals referred to in this chapter.

Planning for Data Communications

Data communications planning should already be complete. The *Network Planning Guide* contains information to help you determine which communications functions to use. The *Physical Planning Guide* contains information to help you plan for communications lines and modems.

Installing Communications Hardware

Communications hardware, such as modems and cables, must be installed before you can start running your programs. (The exception is intrasystem communications, which requires no hardware installation.) However, if your hardware is not yet installed, you can read this manual and begin writing your programs.

Configuring Your System for Data Communications

The *OS/400® Communications Configuration Reference* manual explains how to configure for communications. Although you cannot run your application programs until the system is properly configured, you can read this manual and begin writing your programs.

You need to configure the remote system to allow communications with the AS/400 system. The *OS/400® Communications Configuration Reference* manual contains configuration considerations for some remote systems when communicating with an AS/400 system.

Writing Programs that Use the Intersystem Communications Function (ICF)

You can write communications programs using C/400®, COBOL/400®, FORTRAN/400®, and RPG/400® languages. For an explanation of the communications application interface provided by the intersystem communications function (ICF), read Chapter 3 through Chapter 8. You can then refer to Chapter 9 through Chapter 11 for programming examples that you can use to help write and run programs on the AS/400 system.

You also need the appropriate communications programming manual for the communications type you are using (for example, the *APPC Programmer's Guide*), the programming language manuals for the language you plan to use, and the *DDS Reference* manual.

Operating Communications on the AS/400 System

To use communications on the AS/400 system, you must be familiar with the base operating system as well as the commands unique to communications. Refer to the *Operator's Guide* and the *CL Reference* manual for information on the general operation of the system.

Chapter 2. Communications Features

This chapter introduces the AS/400 system communications features, including:

- ICF communications types
- AS/400 system communications line support
- Base operating system support
- High-level language support
- Additional programming support

Intersystem Communications Function Communications Types

Communications between application programs are accomplished using the AS/400 system ICF and underlying support provided by various **communications types**. Several communications types are provided so that the AS/400 system can communicate with remote systems having different communications methods. Some of the communications methods are:

- Binary synchronous communications (BSC)
- Systems Network Architecture (SNA). Examples of SNA are:
 - Systems Network Architecture upline facility (SNUF)
 - Advanced program-to-program communications (APPC)
 - Finance communications
 - Retail communications
- Asynchronous communications

A communications type, designed for a specific remote system, makes it unnecessary to handle most system-dependent and protocol considerations when coding the AS/400 system application programs. The following communications types are supported by ICF:

- Advanced program-to-program communications (APPC)
- Systems network architecture upline facility (SNUF)
- Binary synchronous communications equivalence link (BSCEL)
- Asynchronous communications
- Intrasystem communications

- Finance communications
- Retail communications

An AS/400 system program uses high-level language operations and communications functions to communicate with a remote system through ICF. A return code, made up of major and minor return codes, informs the program of the success or failure of each operation. You can use several of the communications functions and return codes with any of the communications types. You can use some functions and return codes with only one or two communications types. You can use a program written for use with one communications type, with little or no change, to communicate with a different communications type. The level of change required in the program depends on the two communications types, the communications functions, and the return codes.

Your configuration device descriptions identify the devices on your local system with which communications occur. Each communications type has a corresponding configuration device description of the same type.

AS/400 System Communications Types

The following is a description of the AS/400 system communications types supported by ICF, including a brief overview of the remote systems and devices supported by each type.

Advanced Program-to-Program Communications (APPC): APPC allows the system to communicate with other IBM systems that support the SNA logical unit type 6.2 (LU6.2) architecture. Using APPC allows the system to communicate with:

- Logical unit (LU) own
- Another AS/400 system
- A System/38
- A System/36
- Any other IBM systems (such as the Customer Information Control System for Virtual Storage, CICS/VS) with similar levels of APPC support (refer to the *APPC*

Programmer's Guide for more specific information on supported systems).

APPC allows AS/400 system application programs to start programs on remote systems, and allows remote programs to start programs on the AS/400 system. APPC also allows AS/400 system application programs to start other application programs on the local system. The networking capability of data communications support that routes data in a network between two or more APPC systems that do not need to be adjacent, called **advanced peer-to-peer networking (APPN)**, is available through the APPC interface.

An APPC conversation cannot be used by both the System/38 environment and the AS/400 operating environment. A diagnostic message is sent to an application attempting to open an ICF file in the AS/400 operating environment, to accept a conversation using Common Programming Interface (CPI) Communications, or to open either a communications file or a mixed device file in the System/38 environment for the same APPC conversation. Only one interface can be used for any conversation.

Refer to the *APPC Programmer's Guide* for more information.

Systems Network Architecture Upline

Facility (SNUF): SNUF allows the system to communicate with CICS and Information Management System (IMS) applications on other IBM systems. You can use SNUF to communicate with the following host systems:

- System/370* computer
- System/390* computer
- 30xx
- 43xx

SNUF allows AS/400 system application programs to start programs on remote host systems, and allows programs on remote host systems to start programs on the AS/400 system. Both interactive and batch operations are supported.

SNA 3270 Program Interface: The SNA 3270 program interface allows an AS/400 application to communicate with a host application by sending and receiving 3270 data streams.

Refer to the *SNA Upline Facility Programmer's Guide* for more information.

Binary Synchronous Communication

Equivalence Link (BSC): AS/400 system BSC provides the following:

- Distributed data processing support to the AS/400 system users who want to communicate with another system or device at a remote location using BSC.
- Online and batch communications between application programs on different systems (such as System/38) using BSC.
- Communications with another AS/400 system, System/36, or System/34 using BSC.
- Communications with another AS/400 system, System/36, or System/34, with RPG II support for telecommunications.

BSC allows AS/400 system applications to start programs on remote systems that support BSC, and allows remote programs to start programs on the AS/400 system.

Refer to the *BSC Equivalence Link Programmer's Guide* for more information.

Asynchronous Communications: Asynchronous communications is a method of communications supported by the operating system that allows an exchange of data with a remote device, using either a start-stop line or an X.25 line. The system can use asynchronous communications support to communicate with another asynchronous communications location or with a packet assembler/disassembler (PAD) that gives the system access to an X.25 packet-switching data network (PSDN). The system can use X.25 support to communicate directly through an X.25 network, or to emulate a PAD using the International Telegraph and Telephone Consultative Committee (CCITT) recommendations X.3, X.28, and X.29.

The system can use the AS/400 system asynchronous communications support to communicate with:

- Another AS/400 system
- A System/36
- Asynchronous devices

Asynchronous communications support allows AS/400 system applications programs to start programs on remote systems, and allows remote programs to start programs on the AS/400 system.

Refer to the *Asynchronous Communications Programmer's Guide* for more information.

Intrasystem Communications: Intra-system communications allows communication between two application programs on the same AS/400 system. A source program can acquire more than one session for a given device description, and can have more than one transaction at the same time. However, a source program cannot have a transaction with two different programs on the same session.

Note: Intrasystem communications does not support the concept of a remote system or a remote program. When these terms are used in this manual with regard to intrasystem communications, they refer to the program with which your program is communicating.

Refer to the *Intrasystem Communications Programmer's Guide* for more information.

Finance Communications: Finance communications allows you to attach 3601, 3694, 4701, 4702, 4730, 4731, 4732, 4736, 4737, and Financial Branch System Services (FBSS) controllers to your AS/400 system using **synchronous data link control (SDLC)**. SDLC is a form of communications line control that uses commands to control the transfer of data over a communications line. You can also attach the 4701, 4702, 4737, and Financial Branch System Services controllers using X.25.

Note: 4737 self-service transaction station controllers are configured as FBSS controllers.

In addition, you can attach controllers configured as FBSS controllers using a token-ring or Ethernet local area network (LAN). Since these controllers do not support Ethernet networks, you must use an 8209 bridge when you use an Ethernet configuration on the AS/400 system.

While programs using ICF can communicate with any of the finance controllers, programs that do not use ICF can communicate only with controllers configured as a 3694, 4701, and 4702 on the AS/400 system.

Refer to the *Finance Communications Programmer's Guide* for more information.

Retail Communications: Retail communications allows you to attach retail controllers (3651, 3684, 4680, 4681, 4684, and 4692) to the AS/400 system using the SDLC protocol. X.25 is supported for a 4684 controller, provided it has Retail Industry Programming Support Services (RIPSS) 3.01, a program that provides access to the application files on the 4684 controller.

Note: The 4681 controller is the double-byte character set (DBCS) equivalent of the 4680 controller, and the 4692 is the DBCS equivalent of the 4684 controller. In addition, retail communications allows the AS/400 system to act as an in-store processor in the retail environment.

You can use the AS/400 system in several different retail environments:

- Retail in-store processor environment

You can have a host system such as a System/370 at a remote site with several retail controllers and terminals in your store. The AS/400 system can be installed in your store as an in-store processor to coordinate communications between the host and the retail controllers.

- Retail host processor environment

The AS/400 system can also function as a host system to several retail controllers.

Refer to the *Retail Communications Programmer's Guide* for more information.

Non-Intersystem Communications Function Communications

You can also run non-ICF communications on the AS/400 system, such as the following:

- 3270 device emulation and 3270 BSC application program interface
- Remote job entry (RJE)
- Finance communications
- Transmission Control Protocol/Internet Protocol programs (TCP/IP)
- Common Programming Interface (CPI) Communications

- User-defined communications

Because these communications functions are not part of ICF, they are described in other manuals, which are identified in the list of related manuals in the "Bibliography." Refer also to the *Publications Guide*.

Communicating with Remote Work Stations

No communications programming is required to communicate with remote work stations. The necessary communications programs are provided by the system based on the information provided when the remote work station is configured. The program interface for remote work stations is the same as the program interface for local work stations. Refer to the *Guide to Programming Displays* manual for information on the application interface to remote work stations.

Combinations of Communications Types

You can configure multiple communications device descriptions in the AS/400 system. Multiple **communications configurations**, or the physical placement of communications controllers, the attachment of communications lines, and so on, can be active at the same time. All active configurations do not have to be of the same type. The number of configurations that can be active is determined by the number of communications lines available, and whether any lines are being shared by SNA-type communications. A configuration becomes active when you vary on the configuration, as described in "Varying on Communications Configurations" on page 3-3.

AS/400 System Communications Line Support

The AS/400 system supports the following telecommunication lines (all the lines do not have to be the same):

- Switched point-to-point (manual or automatic answer, manual or automatic call)

- Nonswitched point-to-point
- Nonswitched multipoint
- IBM Token-Ring Local Area Network
- X.25 network
- Ethernet network
- ISDN

Each ICF communications type (except intra-system communications) requires at least one communications line to communicate with a remote system.

Operating System/400

Following is a description of the Operating System/400* (OS/400*) support provided for AS/400 system communications.

Communications Configuration

Before you can use communications on the AS/400 system, you must define the environment through the communications configuration function. This support allows you to create, change, display, and delete the communications network interface, line, controller, and device descriptions.

APPC/APPN requires mode descriptions and class-of-service descriptions. The configuration support provides this function.

An **integrated services digital network (ISDN)**, which is a non-Systems Application Architecture* (SAA*) network that can provide voice, data, and image over the same communications line, requires network interface descriptions and connection lists. The configuration support provides this function.

Refer to the *OS/400* Communications Configuration Reference* manual for more information on communications configuration. APPN support provides the ability to communicate with a remote system without having to manually configure the remote system. Refer to the *APPN Guide* for more information.

Intersystem Communications Function File

The ICF file is used to send and receive data between two application programs, and to describe how to present that data. The ICF file contains the file description identifying the record formats used by the communications application program.

The ICF file allows you to define a single file and the program devices used by that file. An ICF file supports any combination of program devices for all the supported communications types. The application program can then write data to or receive data from any of the program devices defined in the file.

Refer to Chapter 4 for information on creating and using the ICF file.

Data Description Specifications (DDS)

DDS defines the format of the data and the characteristics of the operation used on the data. This information is specified as part of the ICF file, the display file, and the printer file.

Certain DDS functions are unique to communications. These functions are described in Chapter 6. (For general information on coding DDS, refer to the *DDS Reference* manual.)

System-Supplied Formats

System-supplied formats that provide functions similar to those accomplished by using DDS keywords are provided as part of the ICF support, and can be used to do specific communications functions. Refer to Chapter 7 for more information about system-supplied formats.

Control Language

With control language (CL) commands, you can create, change, display, and delete the various communications configurations. A menu interface is also provided to assist you in this function.

ICF file commands are provided that allow you to create, change, and override the file

descriptions. Commands are also provided that allow you to add, change, remove or override device entries for the file. Chapter 4 describes the file commands and their use.

You cannot use CL commands to do ICF communications functions.

For more information on the CL commands for configuring communications, refer to the *OS/400* Communications Configuration Reference* manual.

Security

The security provided on the AS/400 system controls who can use communications device descriptions, and the commands that are used with the device descriptions. Security on both the local and remote systems must be considered in writing and running applications.

See the *Security Concepts and Planning* manual for general system security information and Chapter 8 for communications-specific security considerations.

Error Handling

Major and minor return codes are provided to the application program so that error conditions can be properly handled. Applications written in C/400, COBOL/400 and RPG/400 programming languages can access the return codes to help diagnose problems. In addition, messages are entered in the job log to identify the error that occurred. COBOL/400 and RPG/400 programming languages provide language-defined file status that can be used either in place of, or in addition to, the major and minor return code. C/400 programming language does not have file status values.

You can recover from many communications errors with little or no operator involvement. You may be able to reestablish the session or close and reopen the file to accomplish recovery within the user program. The *Communications Management Guide* describes line errors. Appendix B discusses program error recovery.

High-Level Language Support

You can use AS/400 system communications support to write application programs in the supported high-level languages.

| C/400, COBOL/400, FORTRAN/400, and RPG/400
| programming languages support the ICF interface. Chapter 9 through Chapter 11 provide program examples written in C/400, COBOL/400, and RPG/400 languages.

The programs presented in this manual serve as examples only. They are used to show concepts

and techniques and may not represent the most efficient programming methods.

Additional Programming Support

Support is also provided in addition to the ICF interface to allow the application program to send or retrieve database file members from one system to another. This support is provided by file transfer support (FTS).

Appendix E describes this support.

Chapter 3. Introduction to Intersystem Communications Function

ICF allows program-to-program communications between the AS/400 system and other systems. It also provides program-to-device communications between the AS/400 system and hardware devices.

This chapter provides an introduction to:

- How ICF works
- Some of the terms used to describe ICF
- Configuring for and starting communications
- Defining your ICF file
- How to write a program to use ICF

Notes:

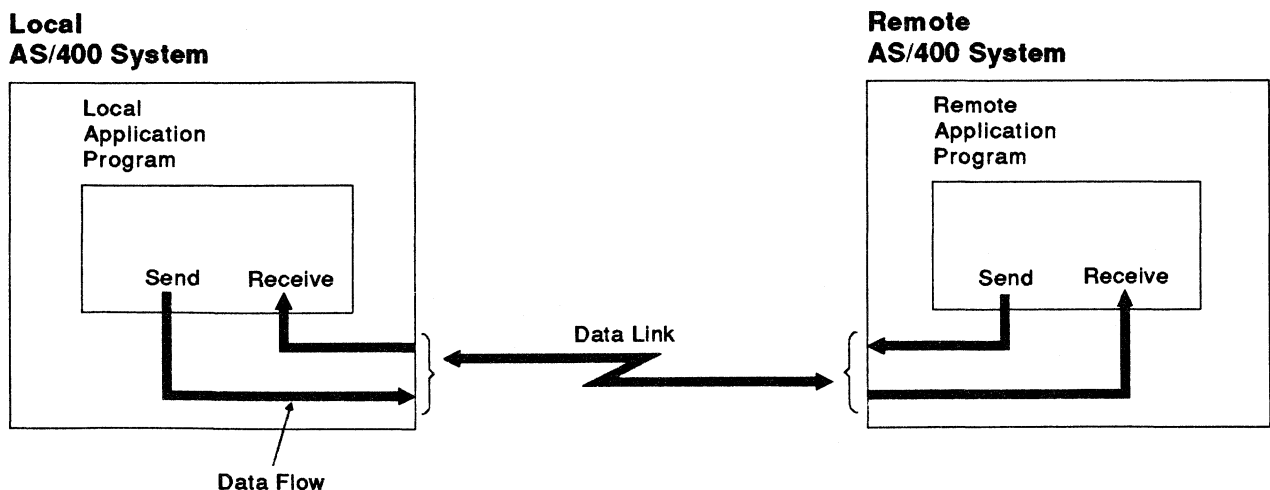
1. The two examples shown in this chapter allow two AS/400 systems to communicate with each other. One program is started by the local AS/400 system operator, which then starts the program on the remote AS/400 system.
2. Although communications types in these examples are advanced program-to-program communications (APPC) and binary synchronous communications equivalence link (BSCCEL), the examples provide a general understanding of how to write a program

that uses any communications type under ICF.

3. Not all communications types require all the operations shown in this chapter. Refer to the appropriate communications programming manual for the communications type you are using for information about a specific communications type.

In Figure 3-1, an application program on the local AS/400 system (local program) sends data to an application program on a remote AS/400 system (remote program) and then receives data.

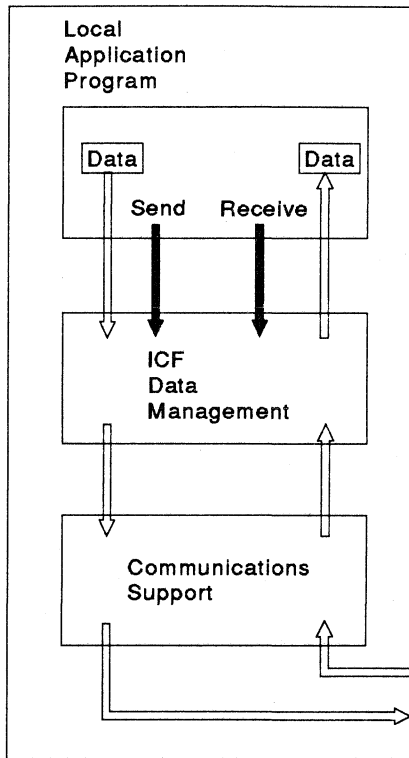
Either program can send data first. You must determine which system is to send data first before you write a program, so you know which operations to do first. Use ICF communications functions and high-level language operations to handle communications within an application program. The ICF functions are described in Chapter 6 and Chapter 7. The language operations you use are the same operations you use when your program is not using ICF. Although these operations are summarized in Chapter 9 through Chapter 11, they are not fully described in this manual. Refer to the appropriate language reference manual for more information.



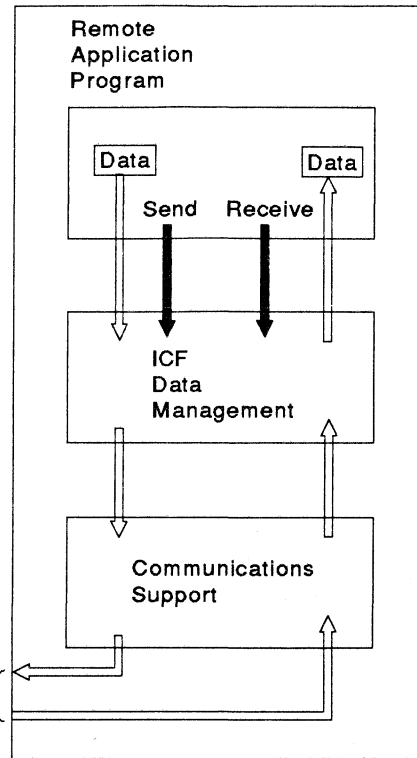
RSL5106-3

Figure 3-1. Sending Data from a Local Program to a Remote Program

Local AS/400 System



Remote AS/400 System



RSLS107-3

Figure 3-2. Major Parts of ICF Data Management

Figure 3-2 shows the major parts of ICF. The **local application program** is the program you write to allow your system to communicate with a remote system. **ICF data management** handles the communications functions and data from your program. The underlying support provided by the communications type handles the communications protocol needed to connect your AS/400 system to the remote system.

ICF data management supports several communications types. Use the communications type that enables you to communicate with your remote system. Refer to Chapter 2 for a list of the communications types and for an overview of the remote systems they support. See the appropriate communications programming manual for a complete description of the remote systems supported by a specific communications type.

Hardware and system-supplied programs handle sending and receiving data on the communications line. Since you do not need to know about these system programs to write an application

program using ICF, these programs and hardware are not described in this manual.

Configuring for Communications

Before communications can occur between two systems, both systems must be configured. You must configure your system to define the appropriate communications hardware and characteristics before you can use your programs. The AS/400 system communications configuration support allows you to create, change, delete, display, and print the following configurations:

- Line descriptions
- Controller descriptions
- Device descriptions
- Mode descriptions
- Class-of-service descriptions
- Network interface descriptions (ISDN only)
- Connection lists (ISDN only)
- Configuration lists

Not all of the listed configurations are used by all the communications types.

Part of the connection between the application and configuration is through the remote location name that is defined as part of the device description. Refer to “The Intersystem Communications Function File” for more information on how this connection is made.

If programs on your system can be started from a remote system, you can define the distribution of work across your subsystems. The AS/400 system considers the communications device to be another source of work for a subsystem. Therefore, you must define a communications entry within the subsystem description to identify the communications devices for which work can be received by the subsystem.

Default communications entries are shipped with the system. However, you can change these entries with the Add, Change, and Remove Communications Entry (ADDCMNE, CHGCMNE, and RMVCMNE) commands. Refer to the *Communications Management Guide* for more information on using these commands. Refer to the *Work Management Guide* for information on subsystems and communications entries.

Varying on Communications Configurations

You must vary on the particular communications configurations you want to use before running your communications applications. (The configurations must already be defined.) The Vary Configuration (VRYCFG) command is used to vary on the appropriate network interface line, controller, and device descriptions.

Note: You can specify that the configurations be automatically varied on at IPL when you create your configurations.

The VRYCFG command does the following:

- Ensures compatibility between the configuration and the communications hardware
- Determines whether the requested data link is available
- Establishes a physical connection with the remote system

Note: For SNA configurations, SNA communications may be established with the

remote system, depending on the line type (switched or nonswitched) and the configuration parameters you have chosen.

The VRYCFG command prepares only the local end of the link to communicate with the remote system. You must also prepare the remote system. Communication can begin when you have prepared both ends and have established a physical connection between the two. For APPC communications, a mode must be started before you establish a session. Generally, the mode starts automatically when the device is varied on or when a request to establish a session is received. You can also use the Start Mode (STRMOD) command to start a mode.

Refer to the *Communications Management Guide* for more information on the VRYCFG command on starting modes.

The Intersystem Communications Function File

An ICF device file defines the layout of the data sent and received between two application programs and links you to the configuration objects that you will use to communicate with the remote system. You identify and use this file in your high-level language application.

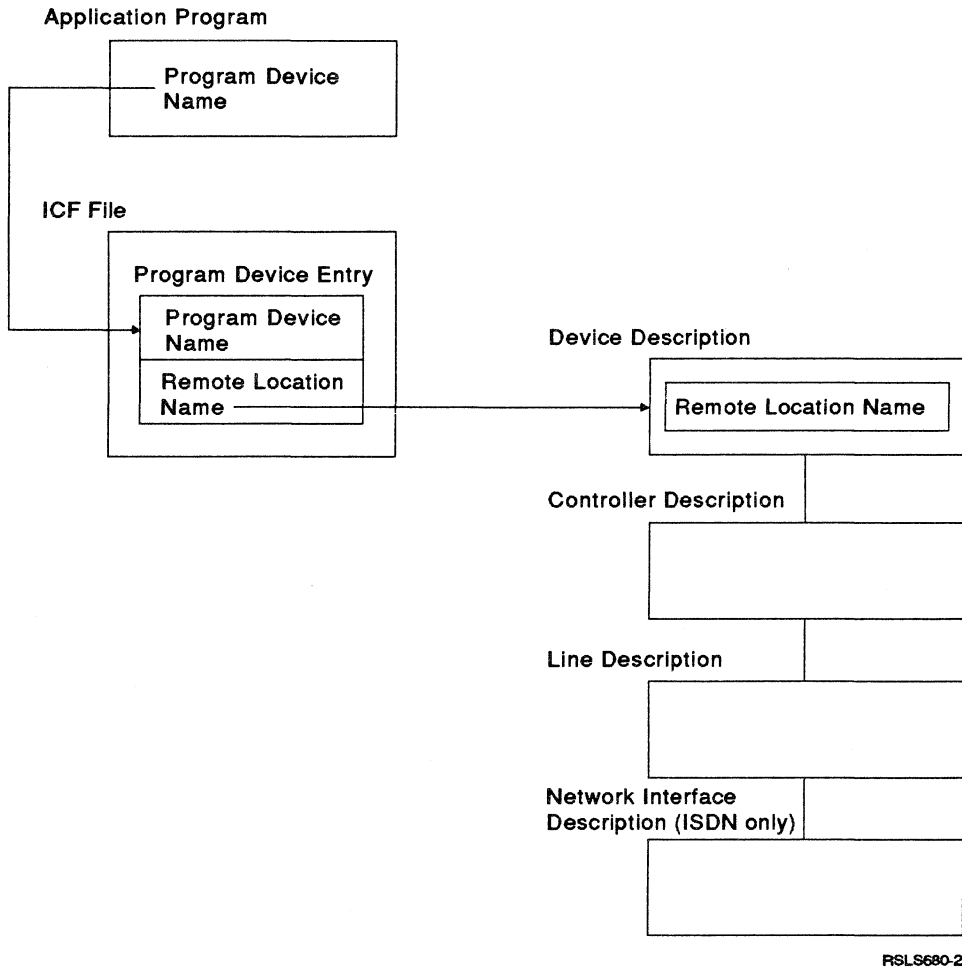
Defining the File

The following commands are used to define the file:

- The Create Intersystem Communications Function File (CRTICFF) command is used to create the ICF file.

Note: If you use system-supplied formats (described in Chapter 7), IBM supplies a file called QICDMF for your use and you do not need to do this step.

- The Add Intersystem Communications Function Device Entry (ADDICFDEVE) or Override Intersystem Communications Function Device Entry (OVRICFDEVE) command is used to define a program device entry. This program device entry is that part of the file that provides the connection to the configuration objects that you will use to communicate with the remote system.



RSL5690-2

Figure 3-3. ICF File-Configuration Relationship

Using the File

An application program uses the file as follows:

- A program communicates through a program device name. The program device name used in the application maps you to the program device entry in the ICF file that contains the same program device name.
- The program device entry also contains a remote location name. This remote location name (specified as part of the device description) provides the final step in completing the link between the application and the device description.

Refer to Chapter 4 for more information on creating the ICF file and on defining program device entries to the ICF file. Also refer to Chapter 4 for more information on the remote location name.

Figure 3-3 shows the relationship between the program, the ICF file, and the communications configurations.

Not all of the communications types require that all of these configurations be explicitly created. Also note that the network interface description is only required when communicating across an ISDN.

Starting Your Program

Your application program can be started by an operator at your system, or by a request from the remote system.

A remote system starts an application program on your local AS/400 system by sending a special record, called a **program start request**, to your system. Refer to "Starting a Program on the Remote System" on page 3-7 for more infor-

information about the program start request. Refer to the appropriate communications programming manual for the communications type you are using for a description of this special record.

Opening the Intersystem Communications Function File

Before communications can occur, your program must open an ICF file (previously created with the CRTICFF command). All communications functions are issued through the ICF file.

Starting Communications with the Remote System

Before your local program can communicate with the remote system, you must establish a **communications session**. A communications session is a logical connection between two systems through which a local program can communicate with a program at a remote location. A communications session is estab-

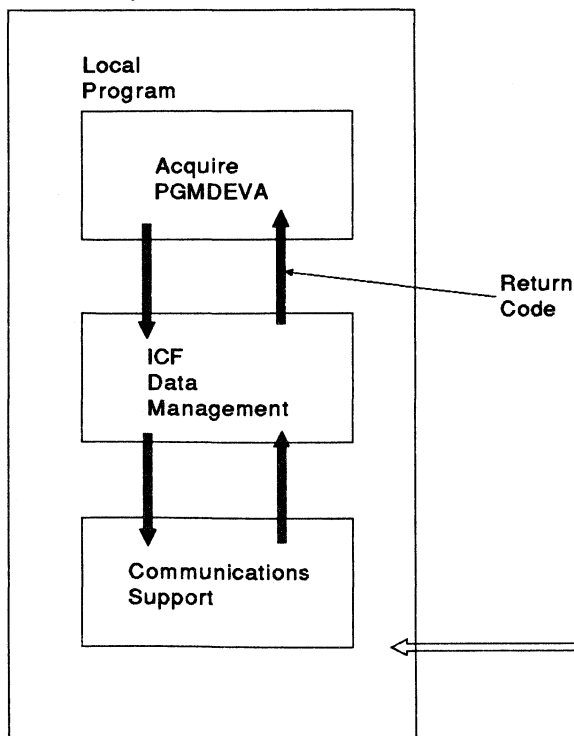
lished with an acquire operation and is ended with a release operation or end-of-session function.

In Figure 3-4, your program establishes a session using an acquire operation with PGMDEVA specified as the program device name.

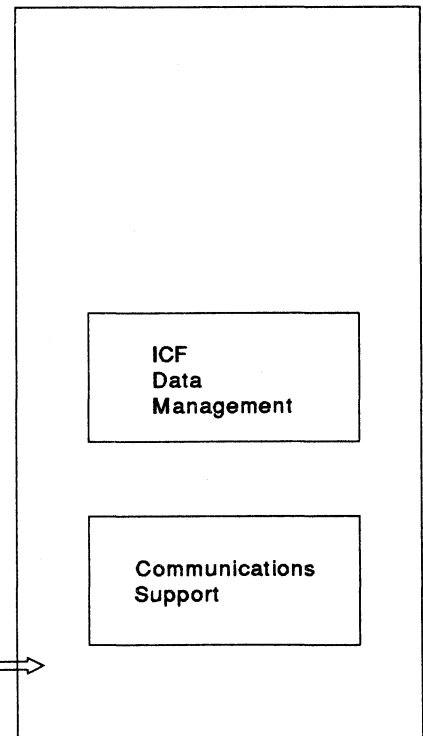
The program device name specified on an acquire operation must correspond to a program device entry in the ICF file with the same program device name. The remote location name associated with that program device entry identifies the remote system with which the session is to be established.

The program device entry is defined with the ADDICFDEVE or OVRICFDEVE command. The PGMDEV parameter specifies the program device name. The RMTLOCNAME parameter specifies the remote location name. The remote location name (also specified as part of the device description) provides the link between the program device entry and the device description.

Local AS/400 System



Remote AS/400 System



Data Link

RSLS110-5

Figure 3-4. Establishing a Session

The following is an example of how a control language program and a high-level language application program are used to acquire a program device. You can use either the ADDICFDEVE or OVRICFDEVE command. This example uses the ADDICFDEVE command.

```

YOURCL
ADDICFDEVE FILE(ICFFILE) PGMDEV(PGMDEVA) RMTLOCNAME(CHICAGO)
      |
      | Identifies the remote
      | location with which
      | your program will
      | communicate.
      |
      | Identifies the name known
      | by the program (PGMDEVA).
      |
      | Identifies the ICF file
      | to which the definition is added.
      |
CALL YOURPROG

YOURPROG
.
.
OPEN ICFFILE
.
.
ACQUIRE PGMDEVA
.
.

```

Note: You can use other parameters with the ADDICFDEVE and OVRICFDEVE commands to

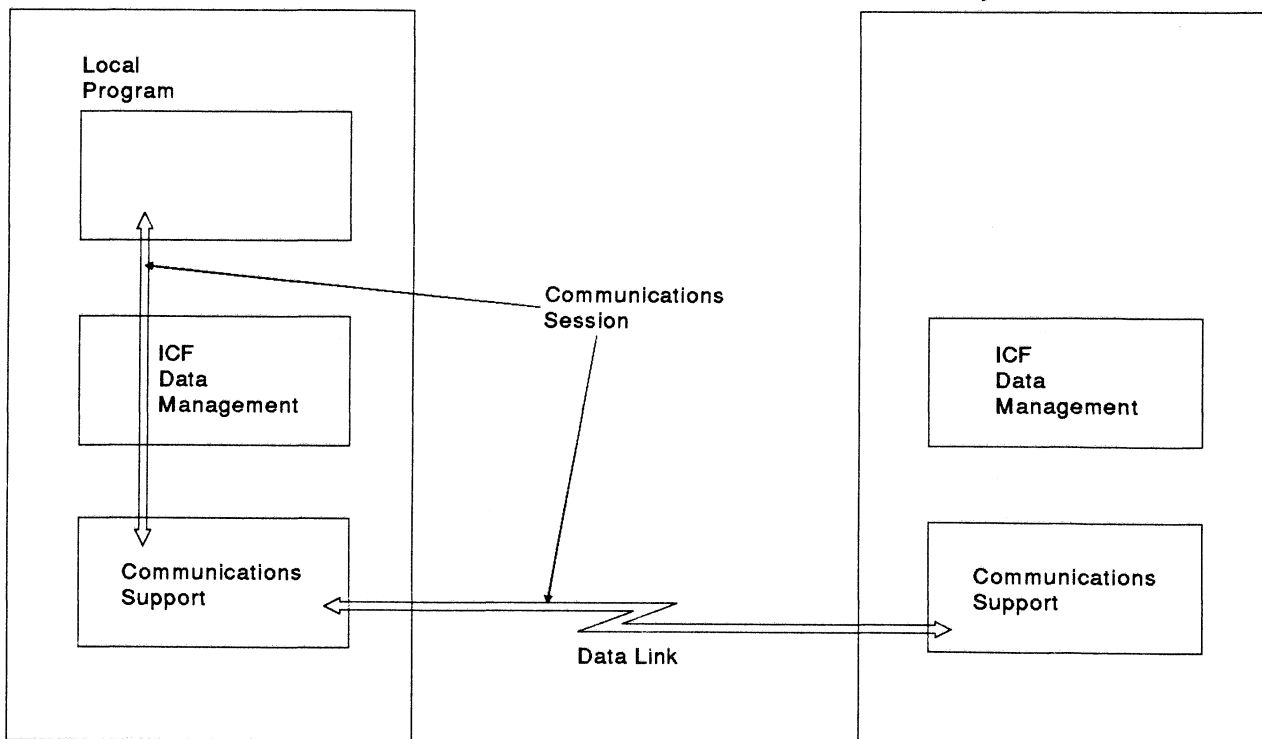
define attributes, such as format selection (FMSTLT), to be used during this session. See Chapter 4 for more information about the ADDICFDEVE and OVRICFDEVE commands and their parameters.

When the program issues an acquire operation, ICF data management returns a return code to your program indicating whether it can communicate (whether a session is established) with the remote system at this time. If communications cannot be established, the return code tells your program why communications failed. See Appendix B for more information about return codes.

Your program cannot send or receive data until the acquire operation succeeds. Therefore, your program must check the return code. In our example, the return code indicates that communications was started. Therefore, a communications session exists between the local AS/400 system and the remote AS/400 system, as shown in Figure 3-5.

Local AS/400 System

Remote AS/400 System



RSLS111-4

Figure 3-5. Communications Session Established

The acquire can be done automatically as a part of the open file operation by specifying the desired program device name (in this example PGMDEVA) on the ACQPGMDEV parameter of the CRTICFF command. Refer to Chapter 4 for more information.

Even though the session has been started, the application program at the remote system has not yet started. "Starting a Program on the Remote System" describes how an application program is started at the remote system.

Starting a Program on the Remote System

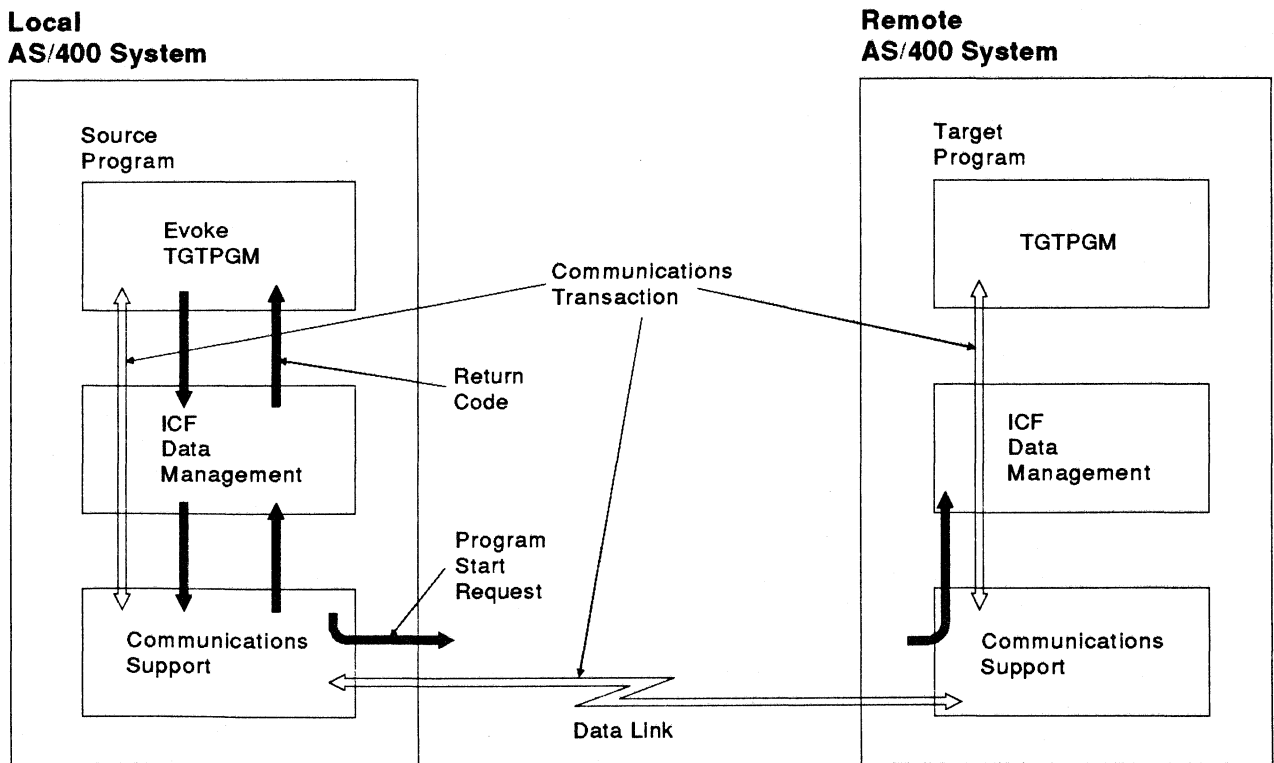
Your program must specify and start the program at the remote system with which it will communicate. After this remote program has been started, a **communications transaction** has been started. A communications transaction is a logical connection between two programs on a session. A communications transaction is started by an evoke function and is ended by a

detach function. After the communications transaction starts, data can be exchanged between the two programs.

Use the evoke function with the necessary parameters to send the name of the program that you want started at the remote system. These parameters include the program name (from either a high-level language program or a control language program), the remote system library where the program is stored, and security information (when required). When your program issues a write operation with the evoke function specified, a program start request is sent to the remote system.

The program that issues the evoke function is the **source program**. The program started on the remote system is the **target program**. In this example, the local program is the source program (it issued the evoke), and the remote program is the target program.

In Figure 3-6, the evoke function is used to start the program named TGTPGM at the remote system.



RSL5112-4

Figure 3-6. Program Started at Remote System by Evoke Function

A return code is always given to your program to indicate the status of the evoke function. In Figure 3-6 on page 3-7, the return code tells your program that the evoke request was accepted and a program start request was sent to the remote system. If the program start request succeeds, the remote system program and the communications transaction start.

Your program can also send program initialization parameters with the evoke function. If the remote system is an AS/400 system, the target program can access any parameters specified with the evoke as if they were parameters passed on a call command.

The type of evoke function you use depends on the communications type you use and on the type of remote system with which you communicate. For more information about the evoke functions, refer to Chapter 6, Chapter 7, and to the appropriate communications programming manual for the communications type you are using.

Connecting to the Session — Target Program

Before a target program can send or receive data, it must first be associated with the session in which the program start request was received. This association is established by opening an ICF file and acquiring a program device associated with a special remote location name of *REQUESTER.

A remote location name of *REQUESTER specifies that:

- The remote location used is the remote location specified in the device description that received the program start request.
- There is no specific remote location assigned to the program device by the ADDICFDEVE or the OVRICFDEVE command.

Any program device name defined in a program device entry with a remote location name of *REQUESTER is referred to as a **requesting program device**.

The target program identifies the requesting program device in the same way that the source program does. The target program specifies, on

an acquire operation, the same program device name as the name specified on the PGMDEV parameter on the ADDICFDEVE or the OVRICFDEVE command.

Figure 3-7 on page 3-9 shows the relationship between the program, the ICF file, and the communications configurations for a requesting program device.

Note: The device description that receives the program start request is the device description that is selected when the acquire operation is issued to the requesting program device.

When the target program issues an acquire operation to the requesting program device, a new session does not start. The acquire only establishes a logical connection between the target program and the session and transaction that were started by the source program.

The remote program cannot send or receive data until the acquire operation is successful.

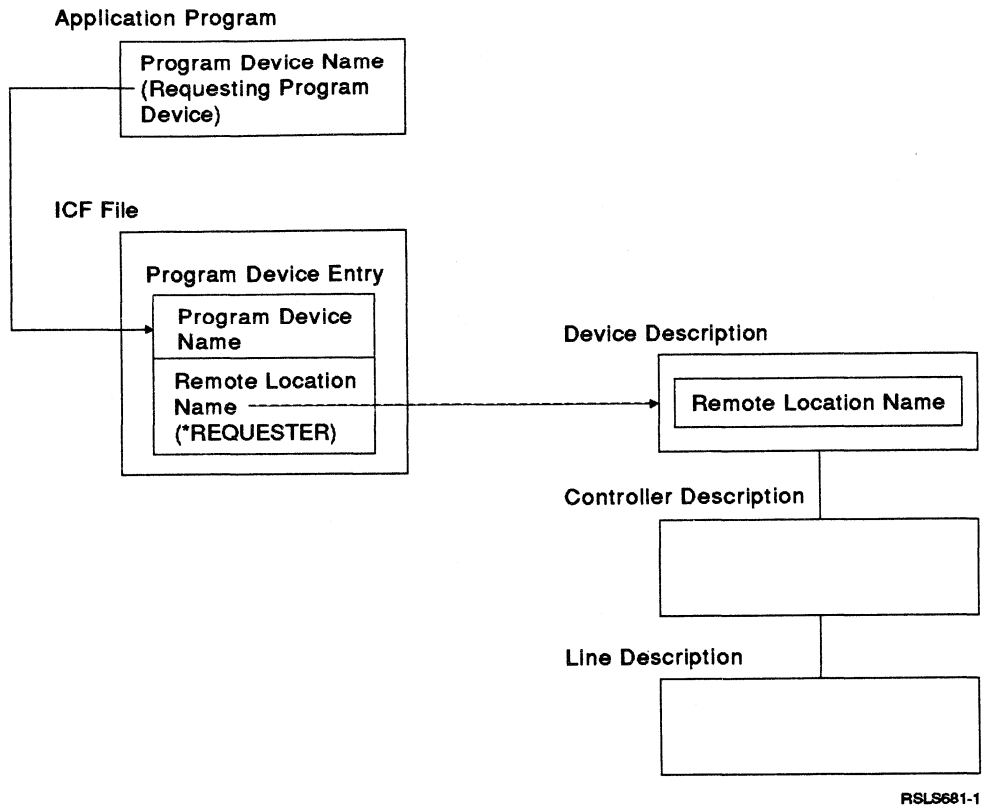
The following shows how a control language program and high-level language application program can be used to acquire a requesting program device.

```
TGTCLPGM
OVRICFDEVE PGMDEV(PGMDEVB) RMTLOCNAME(*REQUESTER)
|
| Identifies that you want to
| communicate with the device
| description that receives the
| program start request.
|
| Identifies the name known
| by the program (PGMDEVB).
```

```
CALL TGTPGM
```

```
TGTPGM
.
.
.
OPEN ICFFILE
.
.
ACQUIRE PGMDEVB
.
.
```

Note: The target program started as a result of a program start request can be a high-level language program or a control language (CL) program. In this example, the CL program containing the OVRICFDEVE command and the call statement is the program that is started as a



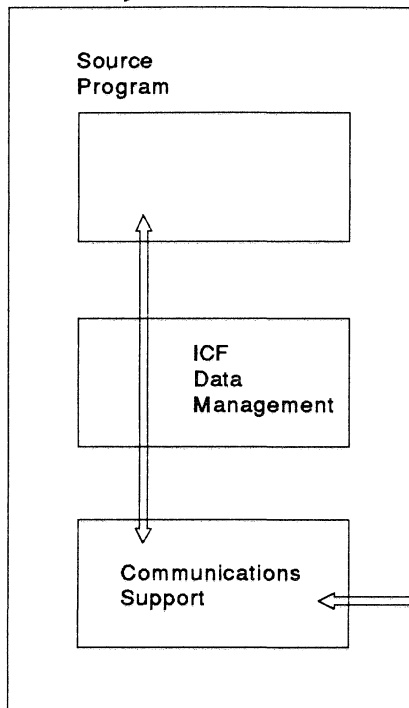
RSL5681-1

Figure 3-7. Requesting Program Device Relationship

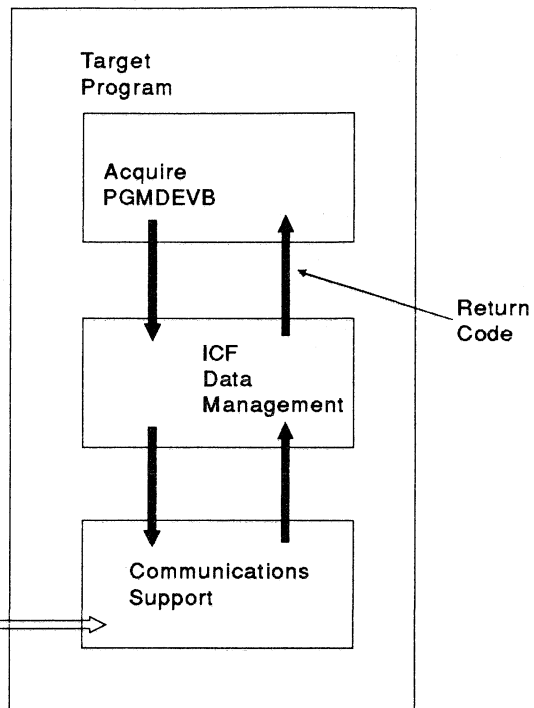
result of the program start request. The CL program calls the high-level language program. In Figure 3-8 on page 3-10, the target program establishes a logical connection to the session

and transaction (started by the source program) by acquiring the requesting program device named PGMDEVB (as assigned by the ADDICFDEVE or OVRICFDEVE command).

**Local
AS/400 System**



**Remote
AS/400 System**



RSL9659-4

Figure 3-8. Establishing a Logical Connection between the Target Program and the Session

The acquire can be done automatically as part of the open file operation, by specifying the requesting program device name (in this

example, PGMDEVB) on the ACQPGMDEV parameter of the CRTICFF command. Refer to Chapter 4 for more information.

Sending and Receiving Data

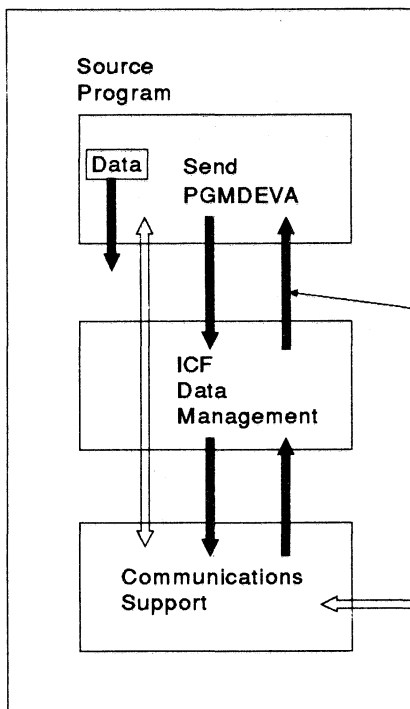
In Figure 3-9, the source program sends data first. To obtain that data, the target program must issue a receive request as the first operation following the acquire.

You use the same program device name specified on the acquire operation on each send and

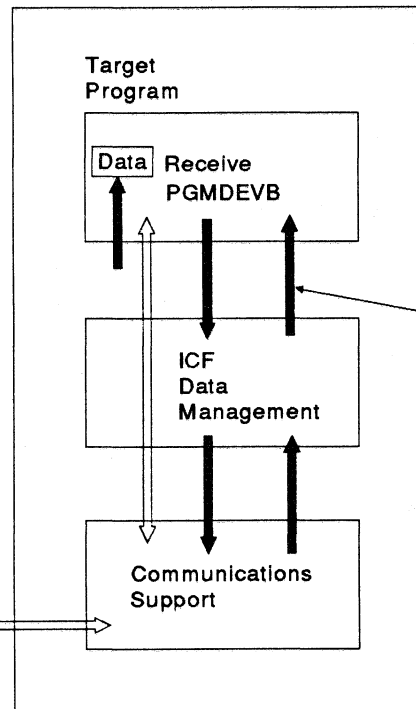
receive request. In the example, the source program uses the program device name PGMDEVA and the target program uses PGMDEVB.

Again, the source program gets a return code indicating the status of the send request. Since the remote system in this example is an AS/400 system, the target program is also given a return code indicating the status of the receive request.

Local AS/400 System



Remote AS/400 System



Return Code

Return Code

Data Link

RSLS113-5

Figure 3-9. Data Sent by a Send Request

Ending Communications with the Remote System

You must end both the communications transaction and the communications session to end communications with the remote system. You can end the communications session in one of the following two ways:

- Explicitly by the program, as shown in Figure 3-10 and Figure 3-11 on page 3-13
- Implicitly ending all sessions and transactions associated with the source program by a close of the ICF file

The transaction and session can be ended by either the source or target program.

Ending the Transaction

The sending and receiving of data continues until one of the two programs ends the communications transaction (either the source or target program can end the transaction). The detach function is used to tell the remote program that

your program has no more data to send and has ended the communications transaction.

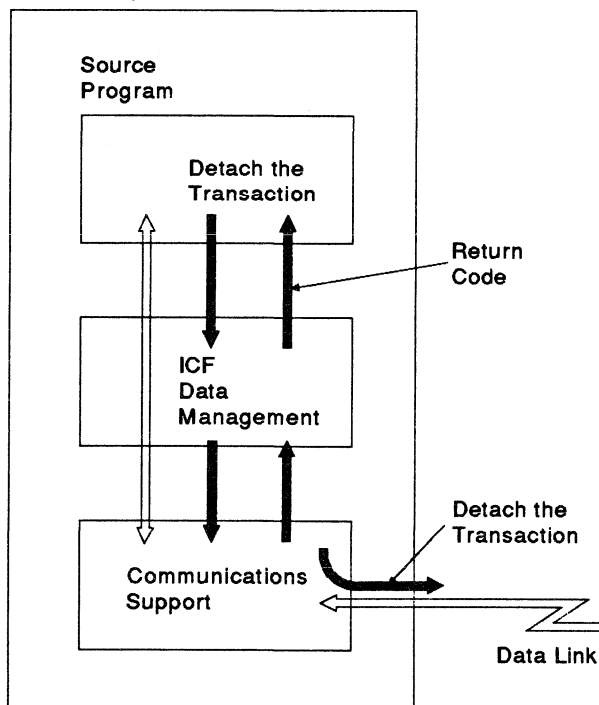
Figure 3-10 shows the source program issuing a detach function to end the communication transaction.

In this example, the target program is given a return code indicating that the transaction has ended. The target program can continue or end processing, but it can no longer communicate with the source program. However, the target program must end the logical connection to the session by ending the session.

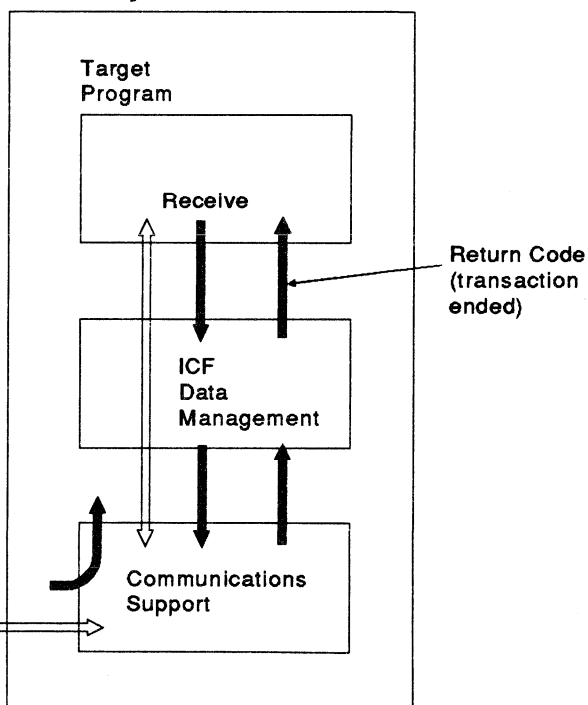
The communications session still exists for the source program. The source program can start another program at the remote system and another transaction, or it can end the communications session and stop communicating with the remote system.

If a target program issues the detach, its logical connection to the session as well as the transaction is ended.

Local AS/400 System



Remote AS/400 System



RSL5114-7

Figure 3-10. Ending a Communications Transaction: Detach Function

Ending the Session

When a session is no longer needed, it should be ended. A source program ends the session by issuing a release operation or end-of-session function. However, a target program must also sever the connection to the session by issuing a release operation or end-of-session function.

Figure 3-11 shows the source program using the release operation and the target program using an end-of-session function to end the session.

When the source program issues the release operation, ICF data management tries to end the session. If the communications transaction has ended, the session ends and the source program receives a return code indicating that the session has ended.

If the session cannot be ended, the source program receives a return code indicating that the release operation was not successful. (For example, the transaction may not have ended.)

If your program cannot recover from the error, you can use the end-of-session function to force the session to end. The end-of-session function always ends the session.

If you issue an end-of-session, you may not be able to determine:

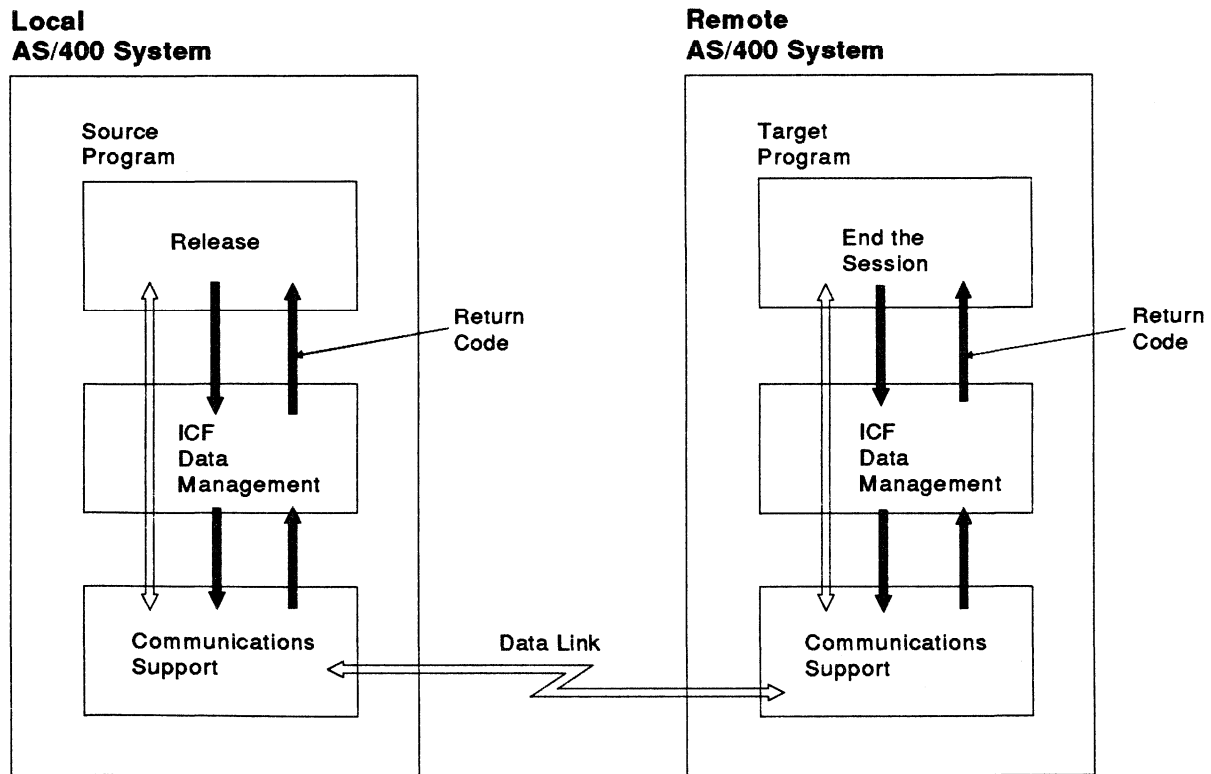
- If the transaction has ended normally
- If all the data has been sent or received

Note: When you use the end-of-session function, your program must make sure all data is received.

The program at the remote system may (depending on the communications type) receive a return code indicating that the session did not end normally.

After ending the transaction and session, a source program can start another session and transaction, continue local processing, or end.

A target program can continue local processing or end after ending a session.



RSL5115-4

Figure 3-11. Ending a Session: Release Operation and End-of-Session Function

Closing the Intersystem Communications Function File

Your program should close the ICF file when you are done processing. Closing the ICF file also implicitly ends any active transactions or sessions for the program.

Varying off Communications Configurations

When you no longer need a communications configuration, you can use the Vary Configuration (VRYCFG) command to vary off the configurations you previously varied on.

If you are using an APPC device, you can end any active modes with the End Mode (ENDMOD) command before you use the VRYCFG command. If you do not use the ENDMOD command, any active modes associated with the

device are ended automatically as part of the VRYCFG.

Refer to the *Communications Management Guide* for information on the VRYCFG and ENDMOD commands.

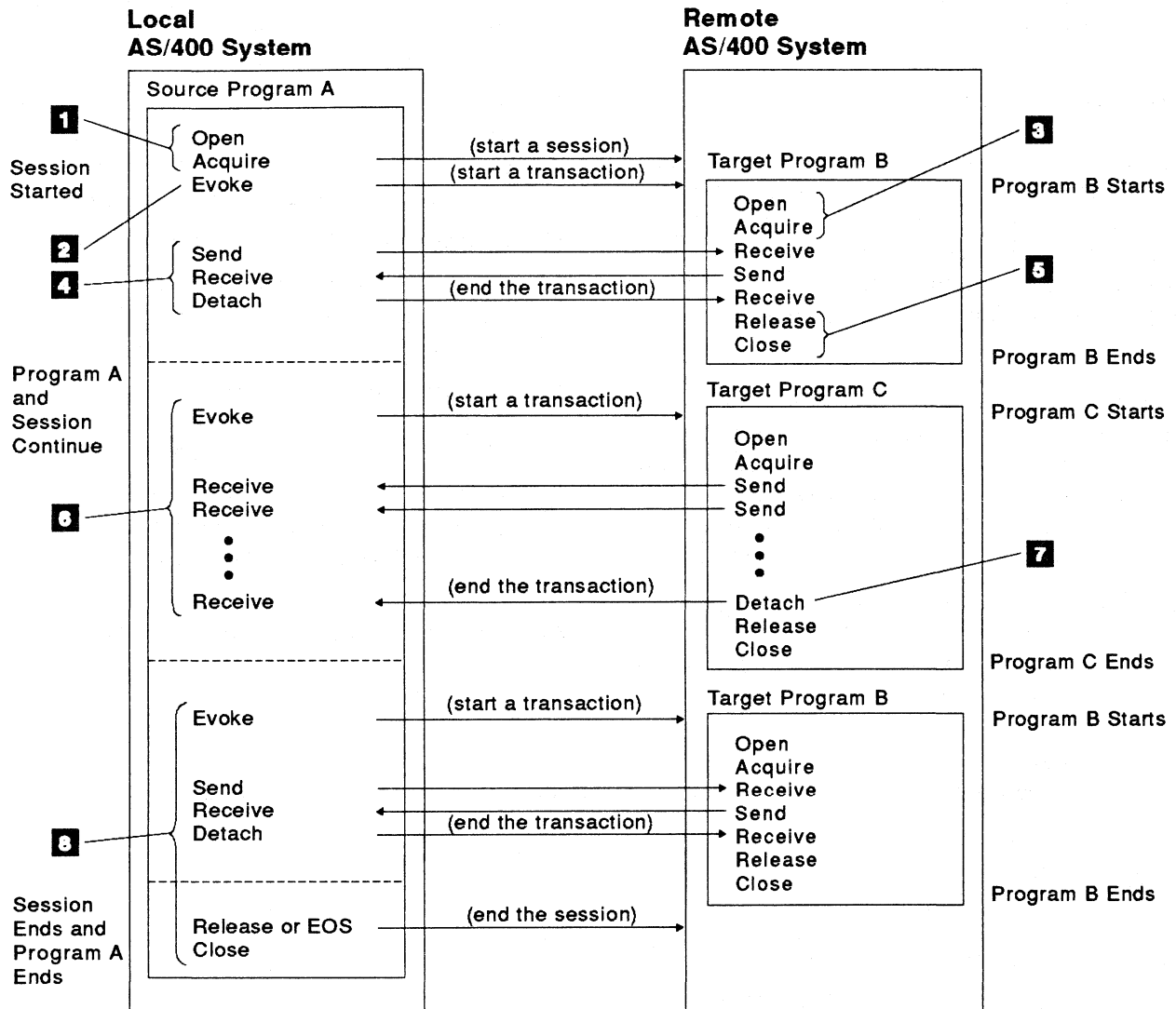
Additional Information on Sessions and Transactions

The information presented in this chapter has only described the flow of two programs using a single session and transaction to communicate with each other.

The following sections describe variations of sessions and transactions.

Multiple Transactions

Figure 3-12 on page 3-15 shows how a source program on a single session can start and end multiple transactions. Only one transaction can be active on a session at a time.



RSL5116-4

Figure 3-12. Starting and Ending Sessions and Transactions

- 1** Program A, on the local AS/400 system, opens the ICF file and then issues an acquire operation to start a session with the remote AS/400 system.
- 2** Program A issues the evoke function, which starts the communications transaction, to start Program B on the remote AS/400 system.
- 3** Program B must open the ICF file on the remote AS/400 system and issue an acquire operation for the requesting program device to establish a logical connection to the session and transaction.
- 4** Programs A and B exchange data. Program A ends the transaction. Program B can end (as shown) or continue processing. Program B cannot, however, communicate with the local AS/400 system.
- 5** Program B releases the session it previously acquired and closes the ICF file.
- 6** Program A starts a transaction with Program C on the remote AS/400 system and exchanges data.
- 7** Program C on the remote AS/400 system ends the transaction. (Either program can end the communications transaction.) Program C releases the session and closes the ICF file.
- 8** Program A starts and ends another transaction with Program B. Program A then releases the session with the remote

AS/400 system, and closes the file on the local AS/400 system.

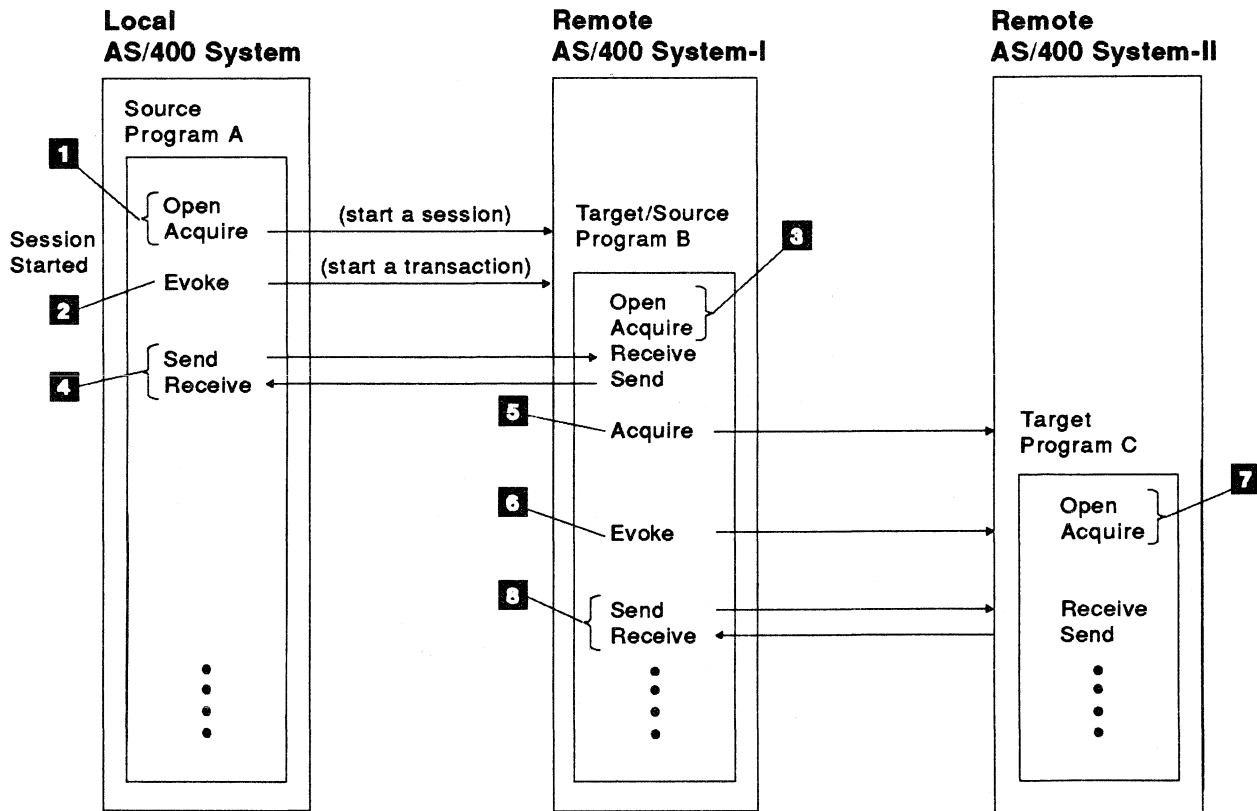
Multiple Sessions

A program can communicate over multiple sessions to the same system or different systems, and can have all the sessions at the same time. When a program is communicating over multiple sessions, it can be both a target and a source program, but it cannot be both on the same session.

A program started by a program start request is the target program for that session. However, this program can also become a source program by establishing a session with another remote system. Figure 3-13 shows how a target program can start a session and a transaction.

- 1 Program A, on the local AS/400 system, opens the ICF file and then issues an acquire operation to start a session with the remote AS/400 system.

- 2 Program A uses the evoke function to start Program B on the remote AS/400 system-I, which starts a communications transaction.
- 3 Program B must open the ICF file and issue an acquire operation for the requesting program device to establish a logical connection with the session and transaction.
- 4 Programs A and B can exchange data.
- 5 Program B issues an acquire operation to start a session with the remote AS/400 system-II.
- 6 Program B uses the evoke function to start Program C on the remote AS/400 system-II, which starts a communications transaction.
- 7 Program C must open the ICF file and issue an acquire operation for the requesting program device to establish a logical connection.
- 8 Programs B and C can exchange data.



RSL5660-5

Figure 3-13. Remotely Started Program Starts a Session and Transaction

Summary

The major tasks you do to use ICF for a source program and a target program are explained in the following sections.

Source Program

Figure 3-14 on page 3-18 shows the sequence of events your AS/400 system application program follows when it starts a session with the remote system.

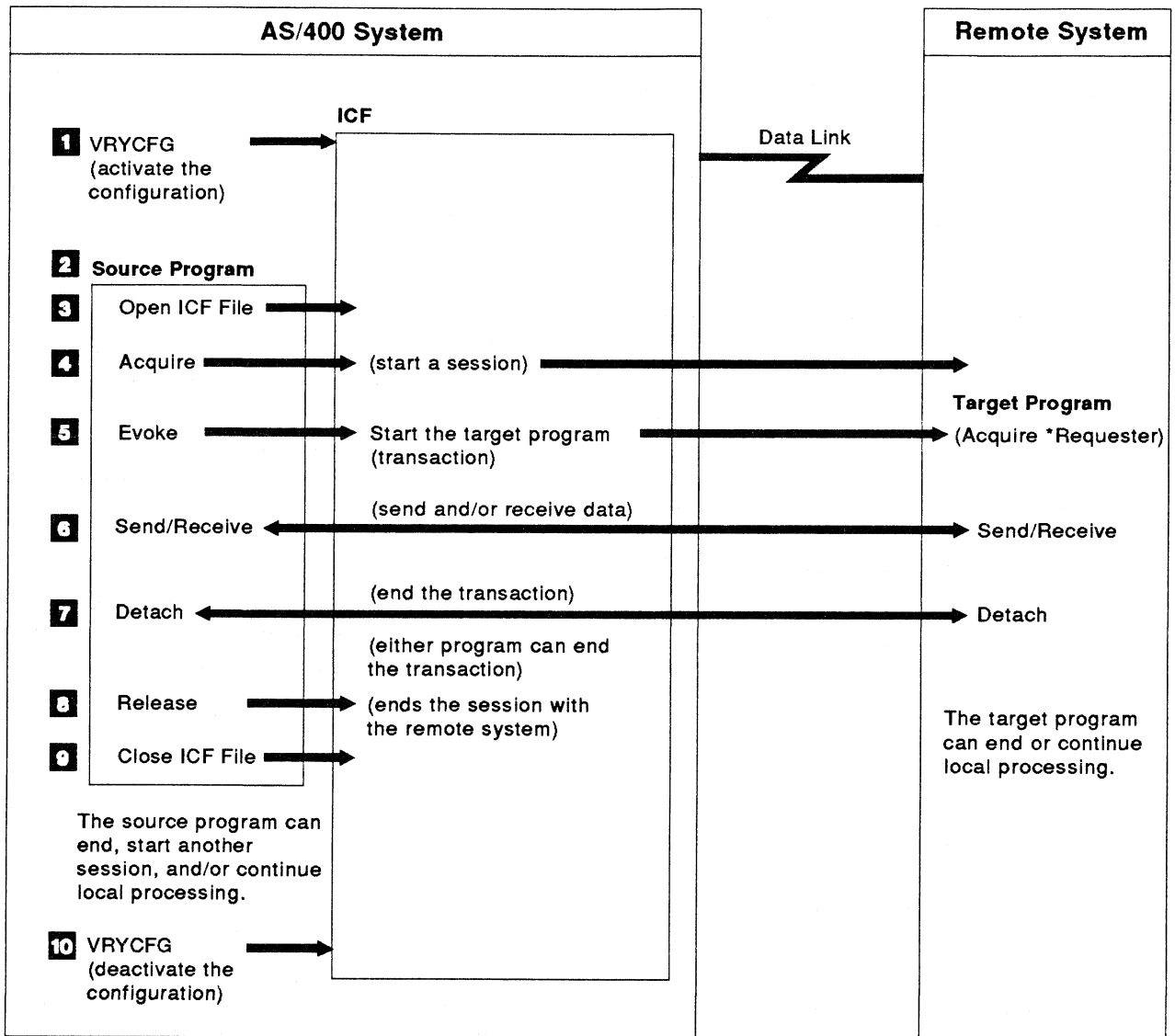
- 1** You must vary on the communications configurations before programs can use them to communicate with a remote system. Use the VRYCFG command to vary on the configurations. You can do the VRYCFG either within the application CL or interactively.
- 2** You must start the AS/400 system application program (source program) that communicates with the program at the remote system.
- 3** The application program must open an ICF file.
- 4** The AS/400 system program must start a session with the remote system before communications can begin. Your program starts a session when it issues an acquire operation.

When your program starts (establishes) the session with an acquire operation, an ADDICFDEVE or OVRICFDEVE command specifies the program device name and the remote location name (identifying the

remote system) associated with the session.

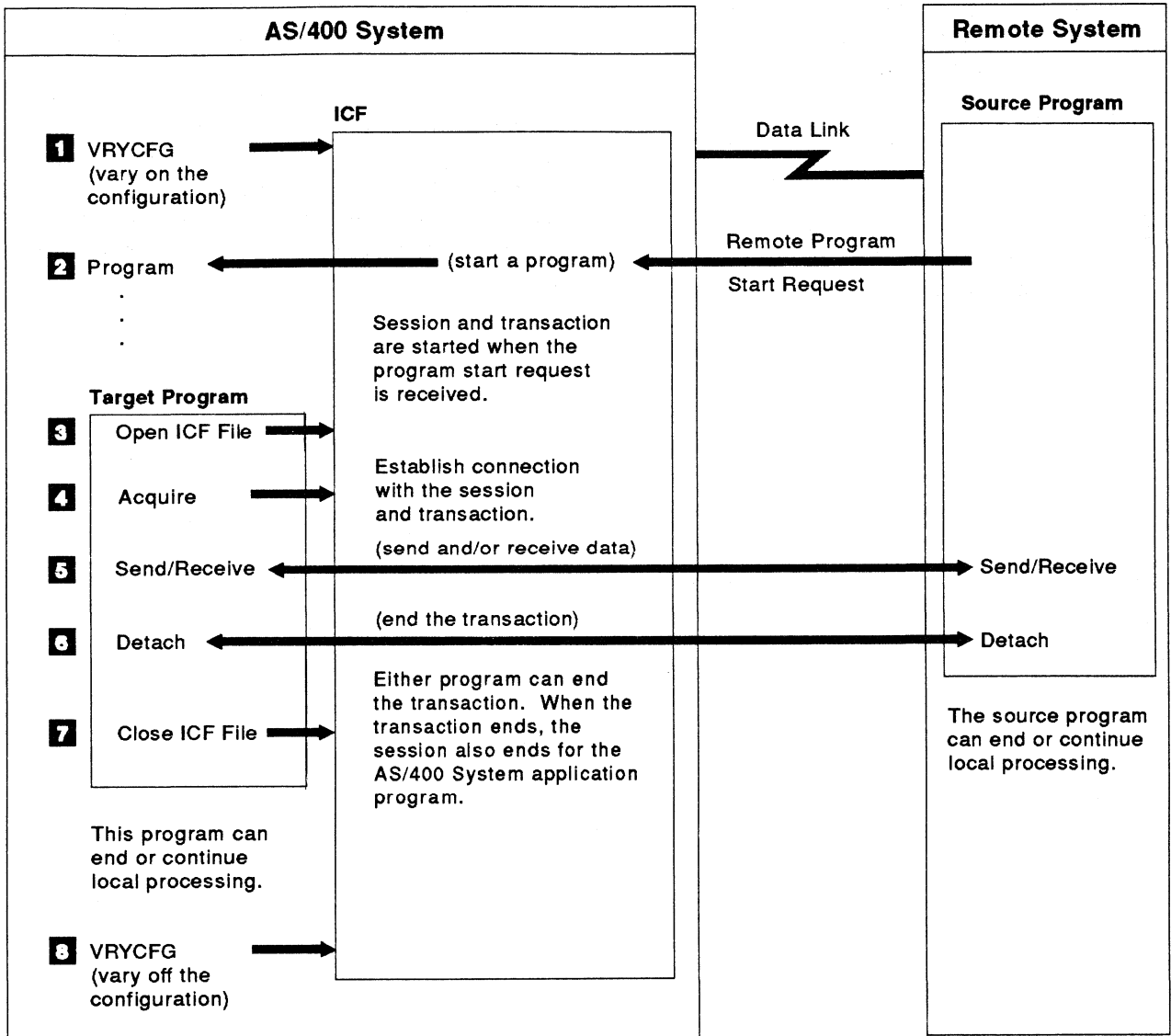
Note: The acquire can be done implicitly as part of the open operation.

- 5** Within each session, you can start (evoke) transactions to allow your program to communicate with target programs. A transaction starts when your program uses the evoke function to start a specified target program.
- 6** Within each transaction, data can be sent and received between the source program and the target.
- 7** Either program can end the transaction when all data has been sent or received. Your program uses the detach function to end the transaction. When the remote system ends the transaction, your program receives a return code indicating that the transaction has ended. If a target program issues the detach, the logical connection to the session is ended implicitly by the detach (a release operation is not needed).
- 8** When all transactions have ended, your program should end the session. Your program can end the session by using the release operation or the end-of-session function.
- 9** Your program must close the ICF file.
- 10** Use the VRYCFG command to vary off the communications configurations when they are no longer needed. You can use the VRYCFG command either within the application CL or interactively.



RSL6102-4

Figure 3-14. The AS/400 System Application Starts a Session with a Remote System



RSLS103-6

Figure 3-15. Remote System Starts a Session with a Program Start Request

Target Program

Figure 3-15 shows the sequence of events that occurs when the remote system starts the session by sending a program start request.

- 1 You must vary on the communications configurations before programs can use them to communicate with a remote system. Use the VRYCFG command to vary on the configurations.

Note: Before your system can process incoming program start requests, you must define subsystem communications entries using the ADDCMNE command. Refer to

the *OS/400* Communications Configuration Reference* manual for more information.

- 2 A program on the AS/400 system is started when your system receives the program start request from the remote system.
- 3 The program must open the ICF file.
- 4 The program must acquire the requesting program device to establish a logical connection to the session and the transaction. The program device name specified on the acquire operation must be associated with a remote location name of *REQUESTER (RMTLOCNAME(*REQUESTER), specified on either the ADDICFDEVE or the OVRICFDEVE command).

- 5** Your program can send or receive data, depending on the procedures previously set up with the remote system.
- 6** Either program can end the transaction when all data has been sent or received.
- 7** Your program must close the ICF file.
- 8** Use the VRYCFG command to vary off the communications configurations when they are no longer needed.

The previous outline summarizes the sequence of events needed for both source and target programs. Overall, every event is required, but different subsets of events can be repeated without repeating the whole series of events. For example, you can acquire and release multiple program devices in the same program. You can also run multiple programs without varying on and varying off the communications configurations.

Chapter 4. Intersystem Communications Function Files

This chapter describes the ICF files, including:

- Using ICF file commands
- Creating and changing ICF files
- Identifying the program devices used with ICF files

Chapter 5 describes how you use ICF files.

Introduction to Intersystem Communications Function Files

A device file is a description of how input data is presented to a program from a device and how output data is presented to a device from a program. A device can be a physical device or a remote system. For example:

- For asynchronous communications, a device can be an ASCII terminal.
- For advanced program-to-program communications (APPC), a device can be a logical unit on a remote system.

Device files do not contain data. Device files contain the file description identifying the device to be used and the record formats used by the application programs. The record formats and associated processing keywords are defined in the data description specifications (DDS) source.

The type of device file used for communications is the ICF file. The ICF file allows the definition of program devices for different communications types. The communications types are advanced program-to-program communications (APPC), Systems Network Architecture upline facility (SNUF), binary synchronous communications equivalence link (BSCSEL), asynchronous, intra-system, finance, and retail communications. Your application program writes data to and reads data from the file. You can specify

whether the data is to be read from a specific program device or from the first program device that responds to a request. You always write data to a specific program device.

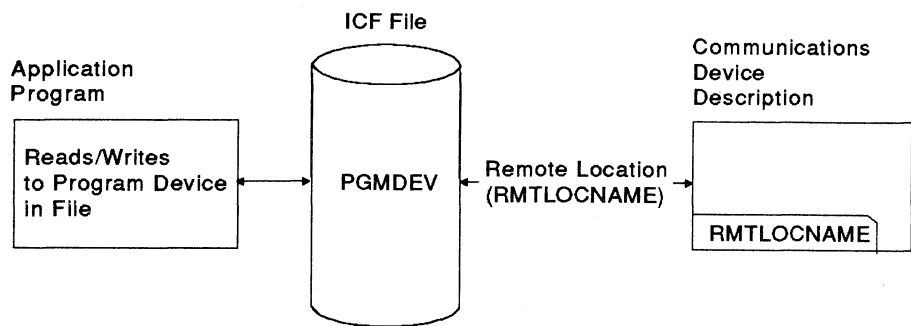
The ICF file allows multiple sessions with different remote systems. You can define and use up to 256 program devices with an ICF file. The program devices can be a combination of different communications types. You must create the necessary communications configuration descriptions for the program devices defined to the file.

Multiple programs (in the same job or separate jobs) can use the same ICF file simultaneously. Each program can have 256 program devices per file.

The file description information for an ICF file is derived from the parameters on the Create Intersystem Communications Function File (CRTICFF) command or the Change Intersystem Communications Function File (CHGICFF) command. The record format information is derived from the DDS that define each record format in the device file and from the fields within each format.

You must also use either the Add Intersystem Communications Function Device Entry (ADDICFDEVE) or the Override Intersystem Communications Function Device Entry (OVRICFDEVE) command to specify the program devices used with the file. These commands provide the connection between the program device name and the remote location name.

The ICF file has attributes unique to ICF and attributes common to other types of device files. Figure 4-1 on page 4-2 provides an overview of ICF files.



RSL8661-6

Figure 4-1. ICF File Overview

Notes:

1. The ICF file is created by using the CRTICFF command.
2. The RMTLOCNAME parameter on the ADDICFDEVE command associates a remote location name to a program device. The remote location name is used to select the appropriate device description.
3. Not all communications types require an explicitly-created device description. For more information, see the appropriate communications programming manual for the communications type you are using.

If you use system-supplied formats (described in Chapter 7), ICF supplies a file for your use. This file is QICDMF in QSYS. If you use this file in your program, you do not need to define DDS or create a file. However, you do need to define the program device entry with the OVRICFDEVE command.

DDS and system-supplied formats provide parallel functions for ICF. System-supplied formats provide a majority of the function without the need to code DDS or to create an ICF file. DDS provides the following additional functions:

- Externally described data
- Additional processing (for example, CONFIRM processing for APPC)
- Indicators to determine session state information
- More flexibility — DDS keywords can be used together in multiple combinations

Intersystem Communications Function File Commands

Three types of commands apply to ICF files: file-level attribute commands, program device entry commands, and commands for displaying information.

File-Level Attribute Commands

The file-level attribute commands are:

Create Intersystem Communications Function File (CRTICFF)

This command creates an ICF file that can be used with communications devices. After the command runs, the file contains the file attributes and the record format definitions.

Change Intersystem Communications Function File (CHGICFF)

This command changes the file attributes of an ICF file.

Override with Intersystem Communications Function File (OVRICFF)

This command can (1) override (replace) the file named in the program, (2) override certain parameters of a file used by the program, or (3) override the file named in the program and override certain parameters of the file to process.

Delete Override (DLTOVR)

This command deletes the effect of the OVRICFF command.

Delete File (DLTF)

This command deletes the file from the system and frees the storage space allocated to that file.

Program Device Entry Commands

The program device entry commands are:

Add Intersystem Communications Function Device Entry (ADDICFDEVE)

This command adds a program device entry with the specified device name and attributes to the file. You can use this command multiple times to add multiple program device entries to the same file.

Change Intersystem Communications Function Device Entry (CHGICFDEVE)

This command changes the program device entry that was defined with the ADDICFDEVE command.

Override Intersystem Communications Function Device Entry (OVRICFDEVE)

This command is used either (1) to override attributes specified in the ADDICFDEVE command or (2) to temporarily associate the specified program device name and attributes to the file. This command is different from the ADDICFDEVE command because it does not permanently change the ICF file. The association between the program device entry and the file is only for the job in which the command runs. You can use this command multiple times to override multiple program device entries to the file.

Delete Override Device Entry (DLTOVRDEVE)

This command deletes the effect of the OVRICFDEVE command.

Remove Intersystem Communications Function Device Entry (RMVICFDEVE)

This command removes one or more program device entries from the file.

Display Information Commands

The commands used to display information are:

Display File Description (DSPFD)

This command displays information about the attributes of a device file.

Display File Field Description (DSPFFD)

This command displays field-level information for a device file.

Display Override (DSPOVR)

This command displays file overrides at any active call level for a job.

Creating an Intersystem Communications Function File

Use the CRTICFF command to create an ICF file. The ICF file contains a file description made up of information specified in two places:

- The source file containing the DDS
- The CRTICFF command

Figure 4-2 shows ICF file creation.

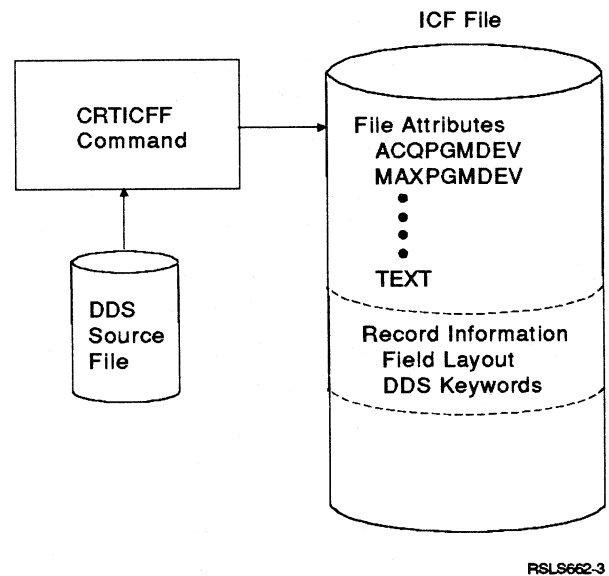


Figure 4-2. Creating an ICF File

Defining the Record Formats for an Intersystem Communications Function File

DDS provides two functions. The first function is to describe the data format as used by the program, by defining record formats and the fields within the records. The second function is to define the characteristics of the operation to be done on the record by the use of DDS keywords. DDS is supplied in the source file specified on the SRCFILE parameter of the CRTICFF file command.

Refer to the *DDS Reference* manual for information on using DDS to define record formats and fields. Chapter 6 describes the function of DDS keywords unique to communications. Information on coding the DDS keywords is in the *DDS Reference* manual.

File Attributes

Figure 4-3 identifies the attributes used with an ICF file. These attributes are specified by the parameters on the CRTICFF command.

Figure 4-3. ICF File Attributes

Parameter	Description
FILE	Name of the file
SRCFILE	Name of the source file containing the DDS
SRCMBR	Name of the member within the source file containing the DDS
OPTION	Output listing options
GENLVL	Severity level of DDS messages that cause the file create to fail
FLAG	Minimum security level of error messages to be listed
ACQPGMDEV	Program device to acquire when the file is opened
MAXPGMDEV	Maximum number of program devices the program can acquire using this file (This parameter also restricts the number of device entries that can be added with the ADDICFDEVE command.)
MAXRCDLN	Maximum record length used with the file
WAITFILE	Length of time to wait for file resources to become available
WAITRCD	Length of time to wait for a record to be returned when performing a read-from-invited-program-devices operation
DTAQ	Name and library of the data queue on which entries are placed
SHARE	Specifies whether the open data path for the file is shared with other opens of the same file in the routing step
LVLCHK	Record format level indicators check
AUT	Default authority granted to the public

Figure 4-3. ICF File Attributes

Parameter	Description
REPLACE	Specifies whether an existing ICF file is replaced
TEXT	Descriptive text describing the file

Acquiring a Program Device when the File Is Opened

Use the acquire program device (ACQPGMDEV) parameter on the CRTICFF command to specify the program device you want to acquire when the file opens. The values for the ACQPGMDEV parameter are described below.

***NONE:** Specifies that no program devices are acquired when the file opens. This value is the default for the ACQPGMDEV parameter.

When you specify *NONE, the program can open the file without consideration of whether the devices to be used are available. In addition, the program does not need a routine to handle errors that occur if the program device cannot be acquired when the file is opened.

The program must acquire at least one program device to the file before doing any input/output (I/O) operations.

Program Device Name: Specifies the name of a program device to be acquired to the file when the file is opened. You can specify the name of any program device associated with the file using the ADDICFDEVE or OVRICFDEVE commands. See "Identifying the Devices Used with an Intersystem Communications Function File" on page 4-8 for more information on how to define a program device to an ICF file. The specified program device must be associated with the file before the file is opened.

When you specify a program device name for the ACQPGMDEV parameter, space is reserved in the file for the specified program device. Refer to the next section for information about reserving space in the file for program devices.

Determining the Maximum Number of

Program Devices: The maximum program device (MAXPGMDEV) parameter on the CRTICFF command specifies the maximum number of program devices you want to use in the file. Use the ADDICFDEVE or the OVRICFDEVE command to associate program devices to the file. Following are guidelines for specifying the value for the MAXPGMDEV parameter:

- An ICF file can either be a single- or a multiple-device file. If your program uses the file as a single-device file, specify a value of 1 on the MAXPGMDEV parameter. If your program uses the file as a multiple-device file, specify the number of program devices simultaneously active to the file. If your file is a single-device file, only one session can be active in the program and the use of the read-from-invited-program-devices operation is restricted. If your file is a multiple-device file, more than one session can be active simultaneously and the read-from-invited-program-devices operation is allowed. Refer to the appropriate language reference manual to learn:
 - How to indicate that the program should use the file as a single- or multiple-device file
 - The differences between single- and multiple-device files
- The value specified on the MAXPGMDEV parameter restricts the number of program device entries you can add to the file using the ADDICFDEVE command.
- The value specified for the MAXPGMDEV parameter indicates the maximum number of program devices you want to have simultaneously active for this file. If you specify a program device name on the ACQPGMDEV parameter, space is reserved in the file for the program device to be acquired when the file opens, and this device must be counted when determining the MAXPGMDEV value. You must, however, still define a program device entry to the file for this program device.

For example, if you specify a program device name of PGMDEVA on the ACQPGMDEV parameter and a 1 on the MAXPGMDEV parameter, the only device that can be

added to the file with the ADDICFDEVE command is PGMDEVA. PGMDEVA is also the only device that can be used with the file. If you specify a 2 for the MAXPGMDEV parameter, you can add and use PGMDEVA and one additional device with the file.

- The value you specify on the MAXPGMDEV parameter should be no larger than necessary. If you specify a larger number of program devices than your program requires, the program uses unnecessary system resources. If the requirements for the maximum number of devices change, you can use the CHGICFF command to change the MAXPGMDEV parameter. Refer to “Changing an Intersystem Communications Function File” on page 4-7 for more information.
- The number of devices a program can handle (while maintaining a reasonable response time) is determined by the amount of processing the program does for each program device.

Determining the Maximum Record

Length: Use the maximum record length (MAXRCDLEN) parameter on the CRTICFF command to specify the maximum record length you want to use with the file. This length is used in calculating the size of allocated I/O buffers and this determines the largest I/O operation that can be performed against the file. The following are guidelines for specifying the value for the MAXRCDLEN parameter:

- If your program uses externally described data and you do not vary the defined length of record formats, use the default value of *CALC. The system then generates the maximum record length based on the largest record defined in the file.
- If you use system-supplied formats in combination with an externally described file, this parameter defines the maximum length you can specify on the system-supplied formats. Since this parameter determines the allocation of I/O buffers and the system rejects output requests that are longer than the allocated I/O buffers, this parameter is important if you try to use a system-supplied format to write a record larger than the largest record in the file. Refer to Chapter 7 for more information about system-supplied formats.

- The value specified on the MAXRCLEN parameter should be no larger than necessary. The value specified can be smaller than the largest record in the file. This can be used to minimize the size of I/O buffers allocated for a program that is using a common file that contains a larger record length than is used by the program.

Determining the Wait-for-File Resources

Value: Use the wait file (WAITFILE) parameter on the CRTICFF command to specify the maximum amount of time the open and acquire operations wait before a file resource (such as a device description) becomes available for use by the file. When a file resource is available to an ICF file, the resource is allocated to the job using that file, and is not available to other jobs.

Some communications types also use the WAITFILE parameter to determine the amount of time to wait for remote communications session resources to become available.

The following are guidelines for specifying the value for the WAITFILE parameter:

- If a session is not available for allocation to the job in which your program is running, the system waits until the session is available or until the specified amount of time elapses.
- If you specify an extremely large value, your program waits a long time before it is notified that the session cannot be established.
- The wait time needs to be increased if your program fails at open or acquire time while trying to acquire a program device to a session that appears to be available.

Determining the Wait-for-Record Value:

Use the wait record (WAITRCD) parameter on the CRTICFF command to specify the maximum number of seconds that the read-from-invited-program-devices operation waits for a response from the invited program devices. Although the normal response is from an invited program device, the read-from-invited-program-devices operation may also complete with a job-being-canceled (controlled) indication.

The value specified for the WAITRCD parameter has no effect on input operations directed to a specific program device. Instead, a read opera-

tion to a specific program device waits until a response is available from that program device.

Refer to Chapter 5 for more information on the read-from-invited-program-devices and read operations.

If your program does not use the read-from-invited-program-devices operation, you need not be concerned about the value specified on this parameter.

The following are guidelines for selecting the WAITRCD parameter value if your program uses the read-from-invited-program-devices operation:

***NOMAX:** Indicates that the read-from-invited-program-devices operation should wait until a response is available from an invited program device. This is the default.

When *NOMAX is specified on the WAITRCD parameter, the read-from-invited-program-devices operation does not return control to the program unless a response is available from an invited program device or the job is ending in a controlled way. If the invited program devices are unable to return a response, the program waits until the job is ended.

***IMMED:** Indicates that the read-from-invited-program-devices operation should not wait for a response from an invited program device.

Specifying *IMMED allows the program to receive a response (if available) from an invited program device. If no response is available, the program receives a 0310 return code without waiting for a time limit to end.

Number of Seconds: Specifies the number of seconds that the read-from-invited-program-devices operation waits for a response from an invited program device. If no response is received from the invited program devices within the specified amount of time, the program is informed through a major/minor return code.

Specifying the number of seconds allows the program to receive a response from an invited program device if a response is available within the specified amount of time. If no response is available within the specified amount of time, the program receives a 0310

return code, indicating that the time limit has ended.

Using a Data Queue: If you want your program to wait for an ICF file and a data queue at the same time, use the data queue (DTAQ) parameter on the CRTICFF command. The program can also wait for a display file if the same data queue is specified on the CRTDSPF, CHGDSPF, or OVRDSPF commands. Refer to “Waiting for a Display File, an ICF File, and a Data Queue” on page 5-17 for more information.

Determining Other CRTICFF Command

Parameter Values: Refer to the CRTICFF command in the *CL Reference* manual to determine the appropriate values for the SRCFILE, SRCMBR, OPTION, GENLVL, FLAG, LVLCHK, SHARE, AUT, REPLACE, and TEXT parameters.

Changing an Intersystem Communications Function File

Use the CHGICFF command to change the file-level attributes of an ICF file. The changes made to the file are system-wide and affect all programs that open the file after the CHGICFF has been done. Any programs that already have opened the file are not affected during the current run. Use the parameters in Figure 4-4 for changing file-level attributes values specified on the CRTICFF command.

Figure 4-4. File Attributes for Changing an ICF File

Parameter	Description
ACQPGMDEV	Program device to be acquired when the file is opened
MAXPGMDEV	Maximum number of program devices that can be acquired by the program using this file; this parameter also restricts the number of device entries that can be added with the ADDICFDEVE command.
MAXRCLEN	Maximum record length used with the file
WAITFILE	Length of time to wait for file resources to become available
WAITRCD	Length of time to wait for a record to be returned when performing a read-from-invited-program-devices operation

Figure 4-4. File Attributes for Changing an ICF File

Parameter	Description
DTAQ	Name and library of the data queue on which entries are placed
SHARE	Specifies whether the open data path for the file is shared with other opens of the same file in the routing step
LVLCHK	Record format level indicators check
TEXT	Descriptive text for describing the file

Overriding an Intersystem Communications Function File

Use the OVRICFF command to temporarily override the file named in the program, the file-level attributes of the file, or both. The OVRICFF command affects only the job in which it is run. Use the parameters in Figure 4-5 for overriding file-level attribute values specified on either the CRTICFF or CHGICFF command.

Figure 4-5. File Attributes for Overriding an ICF File

Parameter	Description
FILE	Name of file to override (same file name as application)
TOFILE	Name of file
ACQPGMDEV	Program device to be acquired when the file is opened
MAXRCLEN	Maximum record length used with the file
WAITFILE	Length of time to wait for file resources to become available
WAITRCD	Length of time to wait for a record to be returned when performing a read-from-invited-program-devices operation
DTAQ	Name and library of the data queue on which entries are placed
SHARE	Specifies whether the open data path for the file is shared with other opens of the same file in the routing step
LVLCHK	Record format level indicators check
TEXT	Descriptive text for describing the file
SECURE	Specifies whether this file is secure from previously called override commands

The scope of the override within the job is determined by its call level. An override command entered at one call level affects all lower call levels. The effects of the override command are deleted when you exit the call level in which the command is entered.

There are two common ways of using the override processing. One way is to do all your override processing for the job at the main call level, as shown in the following example:

```
MAINCLPGM
  OVRICFF FILE(ICFFILE) TOFILE(ICFFILE) WAITRCD(2)
  CALL CPGM
  CALL CBLPGM
  CALL RPGPGM
```

In the example, the OVRICFF applies to all the other programs, because MAINCLPGM is at a higher call level than any of the subsequent programs. The WAITRCD value of 2 seconds is in effect for the programs CPGM, CBLPGM, and RPGPGM.

The second approach is to do your override processing at the lowest possible call level, as shown in the following example:

```
MAINCLPGM
  CALL CCLPGM
  CALL CBLCLPGM
  CALL RPGCLPGM

CCLPGM
  OVRICFF FILE(ICFFILE) TOFILE(ICFFILE) WAITRCD(2)
  CALL CPGM

CBLCLPGM
  OVRICFF FILE(ICFFILE) TOFILE(ICFFILE) WAITRCD(4)
  CALL CBLPGM

RPGCLPGM
  OVRICFF FILE(ICFFILE) TOFILE(ICFFILE) WAITRCD(4)
  CALL RPGPGM
```

When CPGM is called, the WAITRCD value in effect is 2 seconds. The effects of the OVRICFF in CCLPGM are deleted when CCLPGM exits to the MAINCLPGM program. The WAITRCD specified on the CRTICFF is now in effect again. When CBLPGM or RPGPGM is called, the WAITRCD value in effect is 4 seconds.

Refer to the *Data Management Guide* for the rules governing the use of override commands.

Use the DLTOVR command to delete the effect of the OVRICFF command.

You can use the DSPOVR command to display the file override in effect.

The WAITRCD parameter has no meaning in the FORTRAN/400 language, as the FORTRAN/400 language does not allow the read-from-invited-program-device operation.

Identifying the Devices Used with an Intersystem Communications Function File

A program communicates through a program device in an ICF file. A program device entry has two functions:

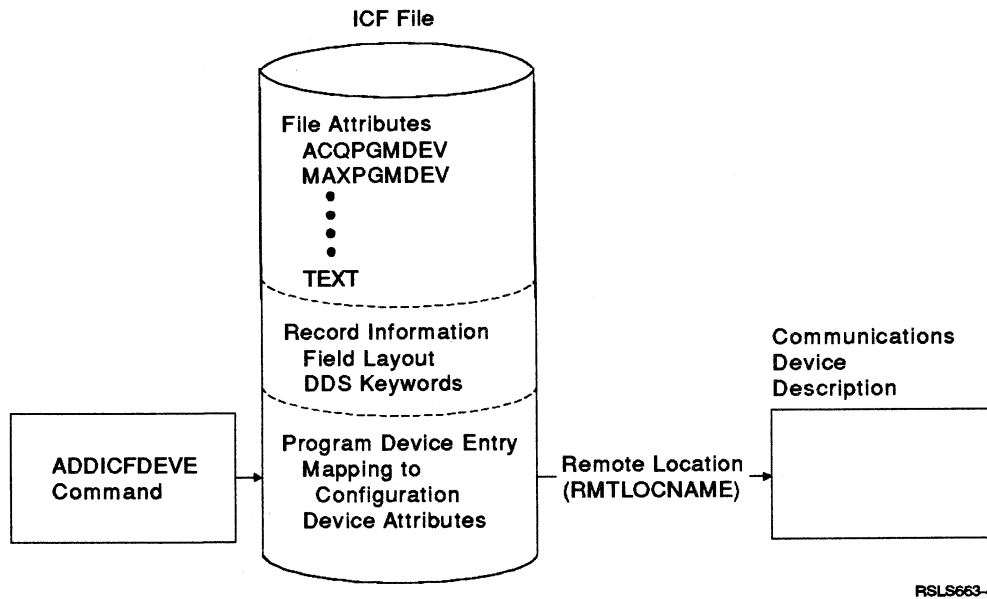
- Associates a program device name with a remote location name
- Establishes a set of program communications-type-dependent attributes

The program device name need not be the same as the name of the device description. To establish an association between the name used in the program (program device name) and the communications configurations, you must define a program device entry to the file.

You must define one program device entry for each name used in the program (even if the name of the configuration is the same as the name used in the program). If the program is written to handle the requesting program device, you must define a program device entry for the requester by specifying a special value of *REQUESTER for the RMTLOCNAME parameter on the ADDICFDEVE or the OVRICFDEVE command.

You must define the program device entry before the program device can be acquired. If you specified a program device on the ACQPGMDEV parameter of the CRTICFF command, you must define the appropriate program device before the file can be opened.

Note: The FORTRAN/400 language does not support program device names. To establish an ICF session using the FORTRAN/400 language, you must use the ACQPGMDEV parameter on the CRTICFF, CHGICFF, or OVRICFF commands. You can have only one ICF session for each ICF file opened using the FORTRAN/400 language.



PSLS663-4

Figure 4-6. Defining a Program Device Entry to an ICF File

Defining Program Device Entries

You can define a program device entry in numerous ways. For example, the ADDICFDEVE command permanently adds the program device entry to the file, while the OVRICFDEVE command provides the same function without changing the file. Figure 4-6 shows how to define a program device entry to an ICF file.

Note: Figure 4-6 shows only one program device entry. Multiple program device entries can be defined. The maximum number of entries is determined by the MAXPGMDEV parameter specified at file creation.

You can define a program device entry to an ICF file at either the file or job levels. A definition at the file level is system-wide and affects all users of the file. A file-level definition adds the program device entry to the specified file. A definition at the job level does not change an ICF file. The definition is only associated with the job for which the command is entered. Because changes made at the job level are not directed to a specific ICF file, all ICF files associated with the job are affected.

File-Level Definition: Use the ADDICFDEVE command to add a program device entry to an ICF file. The program device entry is added to the file specified in the FILE parameter.

Use the CHGICFDEVE command to change a program device entry previously added to an ICF file with the ADDICFDEVE command. The PGMDEV parameter identifies the entry to change. You can use the CHGICFDEVE command to change the association to the communications configurations, the program communications-type-dependent attributes, or both.

Use the RMVICFDEVE command to remove a program device entry previously added to an ICF file with the ADDICFDEVE command. The PGMDEV parameter identifies the entry to remove.

Job-Level Definition: Use the OVRICFDEVE command to override a program device entry defined with an ADDICFDEVE command or to temporarily define a program device entry. You do not have to precede an OVRICFDEVE command with an ADDICFDEVE command.

The OVRICFDEVE command follows the same override processing rules as the OVRICFF command. The following example demonstrates the use of the OVRICFDEVE command at the main call level:

```

MAINCLPGM
OVRICFDEVE PGMDEV(PGMDEVA) RMTLOCNAME(CHICAGO)
CALL CPGM
CALL CBLPGM
CALL FTNPGM
CALL RPGPGM

```

In the example, the OVRICFDEVE applies to the four called programs, because MAINCLPGM is at a higher call level than any of the subsequent programs. Each of the programs, CPGM, CBLPGM, FTNPGM, and RPGPGM, can use PGMDEVA without coding an additional OVRICFDEVE command.

In the following example, the OVRICFDEVE command is used in the lowest call level:

```

MAINCLPGM
CALL CCLPGM
CALL CBLCLPGM
CALL FTNCLPGM
CALL RPGCLPGM

CCLPGM
OVRICFDEVE PGMDEV(PGMDEVA) RMTLOCNAME(CHICAGO)
CALL CPGM

CBLCLPGM
OVRICFDEVE PGMDEV(PGMDEVA) RMTLOCNAME(NEWYORK)
CALL CBLPGM

FTNCLPGM
OVRICFDEVE PGMDEV(PGMDEVA) RMTLOCNAME(NEWYORK)
CALL FTNPGM

RPGCLPGM
OVRICFDEVE PGMDEV(PGMDEVA) RMTLOCNAME(NEWYORK)
CALL RPGPGM

```

Before CPGM is called within CCLPGM, PGMDEVA is associated with RMTLOCNAME(CHICAGO). The effects of the OVRICFDEVE command in CCLPGM are deleted

when CCLPGM exits to the MAINCLPGM program. Before CBLPGM, FTNPGM, or RPGPGM are called within CBLCLPGM, FTNCLPGM, and RPGCLPGM respectively, PGMDEVA is associated with RMTLOCNAME(NEWYORK).

Use the delete override device entry (DLTOVRDEVE) command to delete the effect of the OVRICFDEVE command.

Refer to the *Data Management Guide* for the rules governing the use of override commands.

Mapping Program Device Name to Communications Configurations

The first purpose of the program device entry is to associate a program device name with a device description. This mapping uses the parameters shown in Figure 4-7 on the ADDICFDEVE, CHGICFDEVE, and OVRICFDEVE commands.

PGMDEV

Specifies the program device name being defined (the name used by the program to do operations). The program device name must be unique throughout all entries in the file. You can map two or more different program device names to the same communications configurations. This mapping allows you to have multiple sessions through the same configurations, and to have different device attributes for the same configurations. PGMDEV is a required parameter.

Figure 4-7 (Page 1 of 2). Mapping Parameters for All Communications Types

Parameter	Description	APPC	SNUF	BSCCL	Asyn-chronous	Intra-system	Finance	Retail
PGMDEV	Program device name	X	X	X	X	X	X	X
RMTLOCNAME	Remote location name	X	X	X	X	X	X	X
DEV	Communications device description	X	X			X		

Figure 4-7 (Page 2 of 2). Mapping Parameters for All Communications Types

Parameter	Description	APPC	SNUF	BSCCL	Asyn- chronous	Intra- system	Finance	Retail
LCLLOCNAME	Local location name	X						
MODE	Mode	X						
RMTNETID	Remote network ID	X						

The rest of the parameters are associated with information supplied at various times during configuration. The following descriptions show the relationship of these parameters to the program device definition.

RMTLOCNAME

Specifies the name of the remote location associated with the program device. The remote location name is the primary mapping to communications configurations. A remote location is associated with any device description that contains the same remote location name (RMTLOCNAME parameter on the Create Device XXXX (CRTDEVXXXX) command). For APPC, intrasystem, and SNUF communications, there can be a one-to-many relationship between the remote location name and the device description. For asynchronous, BSCCL, finance, and retail communications, there is a one-to-one relationship.

The remote location name is used by the system to select the device description. For those communications types that support multiple device descriptions per remote location, each communications type defines the criteria for selecting the best device. For a given remote location name, a list of devices (one or more) may be available for use.

Each communications type has specific rules for defining what constitutes an available device. Because asynchronous, BSCCL, finance, and retail communications have a one-to-one relationship between the device and remote location name, no device selection process is necessary. APPC, intrasystem, and SNUF communications all have a means for selecting the best available device for use.

For APPC, intrasystem, and SNUF communications, if you want to use a specific device description instead of allowing the system to select it for you, use the DEV parameter to further qualify the remote location to a specific device description.

Figure 4-8 on page 4-13 shows the relationship of the remote location to the device description.

Note: Your program selects the communications type and communications link by acquiring a program device associated with a remote location name. The system selects a device description, based on availability (such as varied on and not in use), and other parameters that were specified when the program device entry was defined (such as device description). Note that multiple device descriptions can contain the same remote location name. In Figure 4-8 on page 4-13, if your program acquires a program device associated with a remote location of 'A', the system either selects DEV1 or DEV2, and the session established uses the APPC communications type.

For additional information on how the system processes the RMTLOCNAME, DEV, LCLLOCNAME, and RMTNETID parameters for APPC, refer to the *APPC Programmer's Guide*.

The remote location need not exist at the time you define the program device entry. However, the remote location must exist (either as a device description on the system, or as a remote location in the network) when the program device is acquired.

If the communications type you are using allows multiple sessions per remote location,

the same remote location can be mapped to different program device names.

If the program device entry is being defined to process incoming program start requests, the special value of *REQUESTER must be used for the RMTLOCNAME parameter. The remaining parameters on the ADDICFDEVE, OVRICFDEVE, and CHGICFDEVE commands in Figure 4-7 on page 4-10 do not apply (except PGMDEV) and should not be specified.

RMTLOCNAME is a required parameter.

DEV

Further qualifies the remote location to a specific communications device description.

DEV is an optional parameter. If you do not specify the DEV parameter, and there are several communications device descriptions associated with the remote location, the system determines which device to use. Note that the device used may not be the one you want (for example, the device you want may not be varied on).

Note: If you rename a device after specifying the device name and remote location, you must update your ICF file accordingly.

LCLLOCNAME

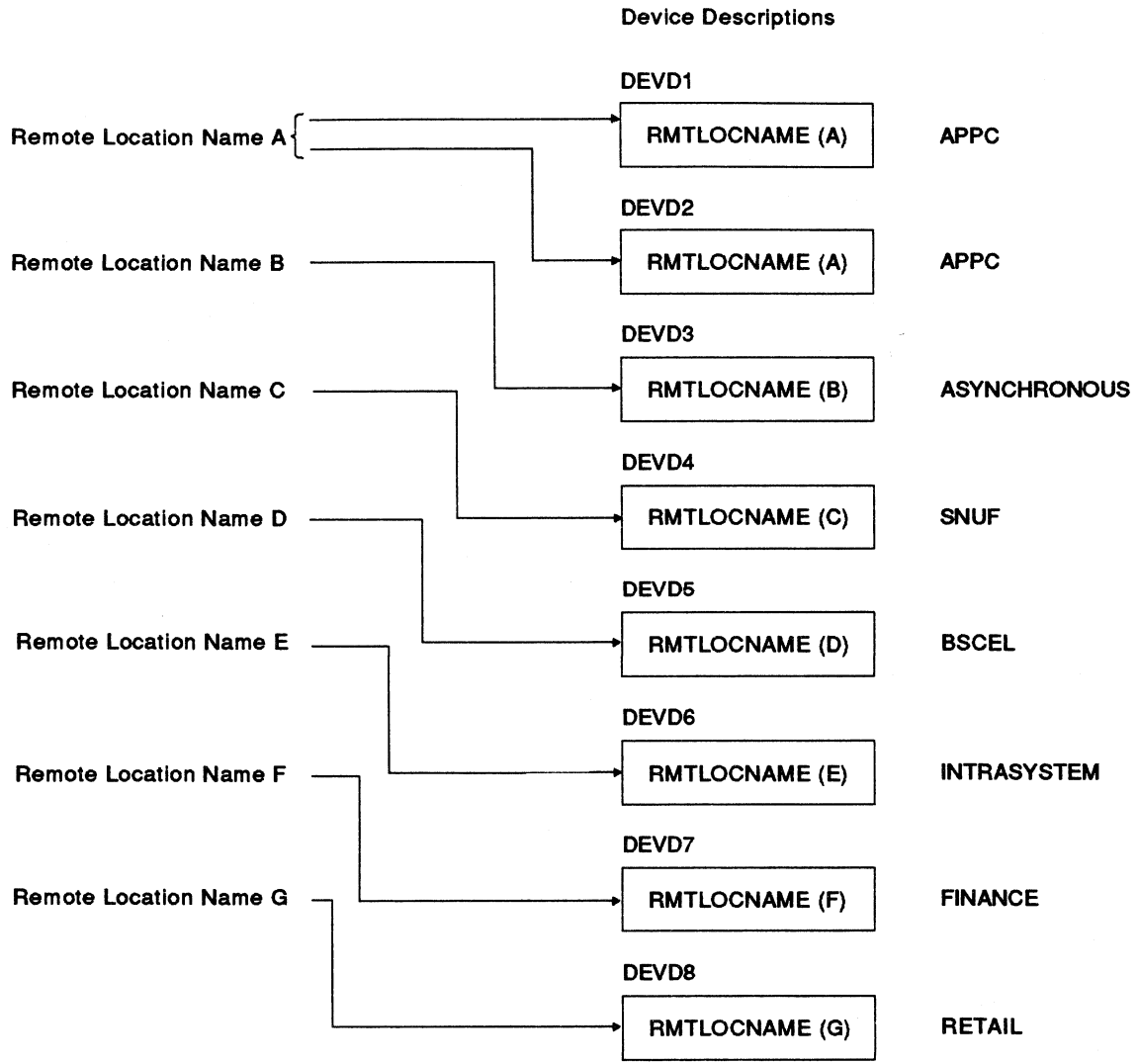
Specifies the local location name of the local system. LCLLOCNAME is an optional parameter.

MODE

Specifies the mode used for the remote location. When you specify the special value *NETATR (the default) the mode in the network attributes is used. BLANK indicates that a mode name consisting of all blanks is used. MODE is an optional parameter.

RMTNETID

Specifies the remote network identifier of the remote location. When you specify the special value *NETATR (the default) the network identifier in the network attributes is used. *NONE indicates that a network identifier consisting of all blanks is used. RMTNETID is an optional parameter.



RSL6682-3

Figure 4-8. Relationship of Remote Location Name to Device Description

If any of the information supplied in the RMTLOCNAME, DEV, LCLLOCNAME, MODE, or RMTNETID parameters conflicts, the acquire operation to the program device fails.

For more specific information, refer to *OS/400* Communications Configuration Reference* manual and the appropriate communications programming manual for the communications type you are using.

Communications-Type-Dependent Attributes

The second purpose of a program device entry is to establish the characteristics of the communications session. These communications-type-dependent attributes are specified as parameters on the ADDICFDEVE, CHGICFDEVE, and OVRICFDEVE commands.

Figure 4-9 describes the communications-type-dependent attributes used with an ICF file and the communication types that support each attribute.

Figure 4-9 (Page 1 of 2). Communications-Dependent Attributes

Parameter	Description	APPC	SNUF	BSCSEL	Asynchronous	Intra-system	Finance	Retail
FMTSLT	Record format selection technique.	X	X	X	X	X	X	X
CMNTYPE	Identifies the communications type you are using to select prompting on the command.	X	X	X	X	X	X	X
APPID	VTAM* identifier of the Customer Information Control (CICS/VS) or the Information Management System for Virtual Storage (IMS/VS) host system.		X					
BATCH	Specifies whether this session is used for batch activity with IMS/VS host system.		X			X		
HOST	Identifies type of host system with which to communicate.		X					
HDRPROC	Specifies whether received function management headers should be passed to the application.		X					
MSGPTC	Specifies whether message protection should be used.		X					
CNVTYPE	Conversation type.	X						

Figure 4-9 (Page 2 of 2). Communications-Dependent Attributes

Parameter	Description	APPC	SNUF	BSCCL	Asyn-chronous	Intra-system	Finance	Retail
BLOCK	Specifies whether system or user will block and unblock transmitted records.			X				
RCDLEN	Maximum record length to transmit and receive.		X	X			X	X
BLKLEN	Maximum block length to transmit and receive.		X	X				
TRNSPY	Specifies whether text transparency is used when sending blocked records.			X				
DTACPR	Specifies whether blanks are compressed when sending and receiving data.			X				
TRUNC	Specifies whether trailing blanks are removed when sending data.			X				
GRPSEP	Specifies separator for groups of data (data sets and documents).			X				
RMTBSCCL	Specifies whether the session supports BSCCL commands and online messages.			X				
INLCNN	Specifies the method of making a connection on the line when a session is established.			X				
SECURE	Specifies whether this program device is secured from previously called override commands.	X	X	X	X	X	X	X

You can specify any parameter on any communications type, but the parameter is ignored if it is not supported by the specified communications type.

Although some attributes, like FMTSLT and CNVTYPE, have system-level defaults, the default

for the majority of the parameters is the information supplied during communications configuration.

Parameters not specified on an ADDICFDEVE, CHGICFDEVE, or OVRICFDEVE command take on the appropriate system default or specified con-

figuration value. If the same parameter is specified on both an ADDICFDEVE and OVRICFDEVE command, the value specified on the OVRICFDEVE overrides the value declared on the ADDICFDEVE command.

The OVRICFDEVE command follows the general rules for override processing. For information on determining the result when two OVRICFDEVE commands are specified for the same program device entry, refer to the *Data Management Guide*.

Format Selection (FMTSLT): The FMTSLT parameter specifies the type of record selection used for input operations for the program device specified in the PGMDEV parameter.

Following are the values for the FMTSLT parameter:

- *PGM**
The record format is determined by the program.
- *RECID**
The record format is based on the use of the RECID DDS keyword.
- *RMTFMT**
The remote program determines the record format to use.

These different values have meaning only when used in conjunction with specific DDS keywords. Refer to Chapter 5 for more information on format selection processing.

Communications Type (CMNTYPE): The CMNTYPE parameter identifies the communications type for which you are defining a program device entry. This identification prompts you for the communications-type-dependent attributes associated with the communications type you are using.

These values are available for the CMNTYPE parameter when *REQUESTER is not specified for the remote location name:

- *ALL**
Prompt for all possible communications-type-dependent attributes
- *APPC**
Prompt for all APPC-supported attributes

- *SNUF**
Prompt for all SNUF-supported attributes
- *BSCSEL**
Prompt for all BSCSEL-supported attributes
- *ASYNCR**
Prompt for all asynchronous communications-supported attributes
- *INTRA**
Prompt for all intrasystem communications-supported attributes
- *FINANCE**
Prompt for all finance communications-supported attributes
- *RETAIL**
Prompt for all retail communications-supported attributes

This parameter is valid only if you enter the command interactively.

However, when you specify *REQUESTER for the remote location name (RMTLOCNAME), you are only prompted for selected values based on the CMNTYPE parameter:

- *ALL**
Prompt for FMTSLT, CNVTYPE, RCDLEN, BLKLEN, and SECURE parameters
- *APPC**
Prompt for FMTSLT, CNVTYPE, and SECURE parameters
- *ASYNCR**
Prompt for the FMTSLT and SECURE parameters
- *SNUF**
Prompt for FMTSLT, RCDLEN, BLKLEN, and SECURE parameters
- *BSCSEL**
Prompt for the FMTSLT and SECURE parameters
- *INTRA**
Prompt for the FMTSLT and SECURE parameters
- *FINANCE**
Prompt for the FMTSLT and SECURE parameters
- *RETAIL**
Prompt for the FMTSLT and SECURE parameters

Note: You can still specify values for the parameters that you are not prompted for (when *REQUESTER is specified for the remote location name) by typing those values and parameters on any command line with any of the program device entry commands. However, the parameter values you specify are ignored and no error return codes are issued.

Secure from Override (SECURE): The SECURE parameter is valid only on the OVRICFDEVE command. This parameter does not apply to the ADDICFDEVE or CHGICFDEVE commands. This parameter is used to restrict the effects of override processing.

Refer to the *Data Management Guide* for information about how the SECURE parameter works.

Refer to the *CL Reference* manual for more information about the format and allowable values.

Other Communications-Type-Dependent Parameters:

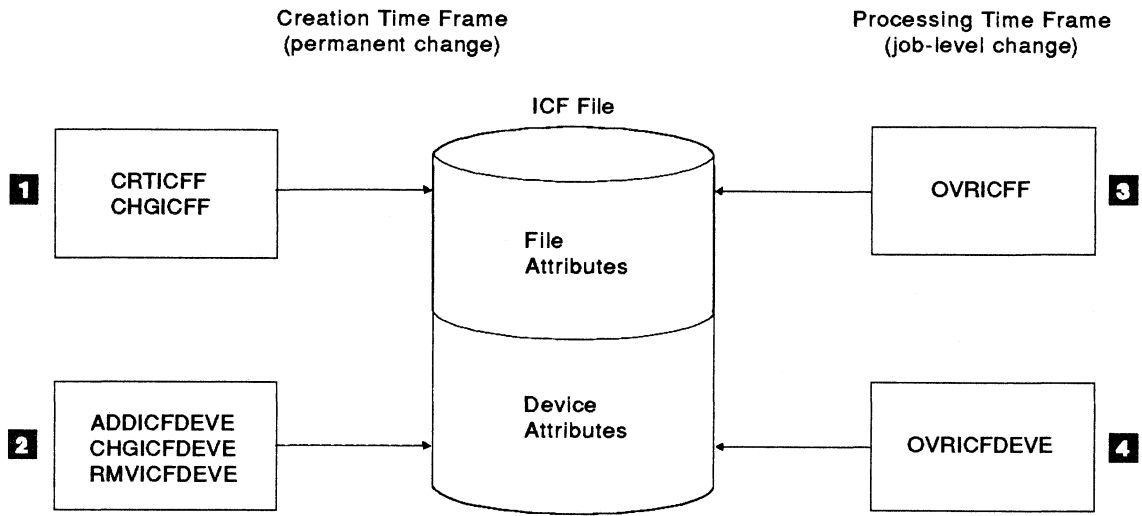
Each of these parameters has specific meaning and function, depending on the communications type you are using. Also, some of these parameters are ignored if the program device being defined is for a RMTLOCNAME(*REQUESTER). Refer to the appropriate communications programming manual for more information on the use of these parameters.

Refer to the *CL Reference* manual for more information on the format and allowable values for these parameters.

Intersystem Communications Function Command Summary

Figure 4-10 on page 4-18 shows the relationship between ICF commands.

- 1** The CRTICFF and CHGICFF commands are used to create the ICF file and work with file-level attributes. (The DLTF command is used to delete the ICF file.)
- 2** The ADDICFDEVE, CHGICFDEVE, and RMVICFDEVE commands allow defining program device- or communications-type-dependent information in the ICF file. The ADDICFDEVE command is optional, and is used only to support early binding, and setting of system-wide defaults.
- 3** The OVRICFF command allows the changing of file-level attributes at the job level only. This command does not cause any permanent change, and it is not system-wide. (The DSPOVR command displays the information entered on the OVRICFF command. The DLTOVR command is used to delete the effects of the OVRICFF command.)
- 4** The OVRICFDEVE command affects the processing of the ICF file at the job level only. You do not specify the file on the override. Whatever ICF file is active at the time is the file that is overridden. It does not cause any permanent change, and it is not system wide. The OVRICFDEVE command uses a late binding function that ties the program device and the remote location at job time. The OVRICFDEVE command is also used to temporarily change communications-type-dependent attributes. The OVRICFDEVE command is job-wide, and has the characteristics of an override command. (The DLTOVRDEVE command is used to delete the effects of the OVRICFDEVE command.)



RSL5664-4

Figure 4-10. Relationship between ICF Commands

Chapter 5. Using an Intersystem Communications Function File

This chapter describes how an application uses an ICF file.

To use an ICF file, identify it as a WORKSTN file in RPG/400 programming language or as a TRANSACTION file in COBOL/400 programming language. For C/400 programming language, the type of file need not be specified.

C/400, COBOL/400, and RPG/400 programming languages support an interface that allows the application to perform the following operations:

- Open the file
- Acquire a program device
- Read from a program device
- Write to a program device
- Release a program device
- Close the file

The FORTRAN/400 language supports an interface that allows the application to perform the following operations:

- Open the file
- Read from the file (program device is implied)
- Write to the file (program device is implied)
- Close the file

Read and write operations are done using a record that contains data description specifications (DDS) keywords. These DDS keywords allow more specific communications functions to be done with the read and write operations. ICF also supports system-supplied record formats that can be used in place of user-defined DDS record formats. Refer to Chapter 6 and Chapter 7 for more information about the communications functions that can be performed with the read and write operations.

Sample programs in Chapter 9 through Chapter 11 provide an overview of the language interface that supports these functions. For more information on the language interface, refer to the appropriate language reference manual.

Opening an Intersystem Communications Function File

The processing done by the open operation depends on whether the open is a subsequent open of a shared file. The open is a subsequent open of a shared file if you specify SHARE(*YES) and the file is currently open with SHARE(*YES) specified.

If the open is not a subsequent open of a shared file in the same job, the open operation allocates the file and any other resources needed to support the acquiring of program devices to the file, and allocates the input/output (I/O) buffers.

If the open is a subsequent open of a shared file, the program is simply attached to the already open file. Any program devices acquired by other programs are available for use by this program. The state or attributes of the file do not change during a subsequent open. For example, if the program device specified as the ACQPGMDEV parameter has been released, the subsequent open does not cause it to be acquired.

After the file and other resources have been allocated, the open operation implicitly acquires the program device specified by the ACQPGMDEV parameter, on the Create Intersystem Communications Function File (CRTICFF), Change Intersystem Communications Function File (CHGICFF), or Override Intersystem Communications Function File (OVRICFF) command. See "Acquiring a Program Device when the File Is Opened" on page 4-4 for information on how to specify the ACQPGMDEV parameter.

The following is a description of the processing done by the open operation based on the ACQPGMDEV parameter value specified for the file:

- If you did not specify the ACQPGMDEV parameter, or if you specified *NONE, the open operation does not acquire any program devices. A program device must be explicitly acquired for the file before the program tries any I/O operations to the file, or before another program opens the same

file if the file is opened with SHARE(*YES) specified.

- If you specify a program device name on the ACQPGMDEV parameter, the open operation acquires the specified program device. See “Acquiring a Program Device” for information about acquiring a specific program device.

If the open operation is not successful, the only allowable operation is closing the file. See “Closing an Intersystem Communications Function File” on page 5-19 for more information.

Obtaining Information about the Open Intersystem Communications Function File

After the program opens the file, an open feedback area is available to the program. This area contains information about the open file such as the file name, library name, and program device information. You can use the information in this area as long as the file is open. See Appendix C for a summary chart of the open feedback fields. Refer to the appropriate language reference manual for information on accessing the fields.

Acquiring a Program Device

Before any input or output operations can be directed to a program device, the program device must be acquired.

Only program devices defined to the file by use

of the Add Intersystem Communications Function Device Entry (ADDICFDEVE) or the Override Intersystem Communications Function Device Entry (OVRICFDEVE) command can be acquired. See “Identifying the Devices Used with an Intersystem Communications Function File” on page 4-8 for more information about defining program device entries.

A program device can be acquired in two ways:

- One program device can be implicitly acquired through the open operation. Refer to “Opening an Intersystem Communications Function File” on page 5-1 for more information on an implicit acquire through the open operation.
- A program device can be explicitly acquired through the acquire operation. The acquire operation can be used many times with different program device names.

When a program device is explicitly acquired with an acquire operation, you identify the session to establish by using the same program device name on the acquire as specified on the PGMDEV parameter on the ADDICFDEVE or the OVRICFDEVE command.

The examples in Figure 5-1 on page 5-3 show the relationship between the program device entry (defined using an ADDICFDEVE or an OVRICFDEVE command) and a C/400, COBOL/400, and RPG/400 operation.

Example 1

```
ADDICFDEVE FILE(ICFFILE) PGMDEV(PGMDEVA) RMTLOCNAME(CHICAGO)
```


RPG/400 Program
'PGMDEVA' ACQ ICFFILE



Example 2

```
OVRICFDEVE PGMDEV(PGMDEVA) RMTLOCNAME(CHICAGO)
```

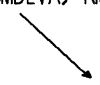
COBOL/400 Program
ACQUIRE 'PGMDEVA' FOR ICFFILE.



Example 3

```
OVRICFDEVE PGMDEV(PGMDEVA) RMTLOCNAME(CHICAGO)
```

C/400 Program
XRacquire(FP,"PGMDEVA")
OPEN(10, ACTION='READ/WRITE', FILE='ICFFILE')



RSL5665-7

Figure 5-1. Relationship of Program Device Entries to Operations

Note: For FORTRAN/400 programs, the acquire operation can be done implicitly by the system only when the ICF file is opened by specifying a program device name on the ACQPGMDEV parameter of the CRTICFF, CHGICFF, or OVRICFF command. The FORTRAN/400 language does not support program device names for ICF files; the program device is implied on read and write operations.

Acquiring the program device automatically allows any I/O operations valid for that program device to be issued. For example, if the file is opened for input only, the read operation is allowed, but the write operation is not allowed.

The amount of time the system waits for resources to become available to complete the acquire request is specified on the WAITFILE parameter of the CRTICFF, CHGICFF, or OVRICFF command.

The following sections describe some of the functions performed when a program device is acquired.

Acquiring a Program Device — Source Program

The system tries to allocate a new session with the remote location for the job in which the program is running.

Some causes of a failed acquire operation are:

- The device associated with the program device is not varied on.
- The device description for the device associated with the program device is allocated to another job.
- A session is not available for the remote location.

Acquiring a Program Device — Target Program

The system tries to establish a connection with the requesting program device. An acquire operation to the requesting program device does not allocate a new session. It only establishes a logical connection to the session and transaction on which the target program was started.

The acquire operation fails if any of the following occur:

- The session was previously ended by the target program.
- A program not started by an evoke function issues an acquire for a requesting program device.
- The requesting program device is acquired for another file in the job.

You can use the information in the program device definition list as long as the program device is acquired. The support provided by the high-level language you use determines whether you can access this information.

See Appendix C for a summary chart of the program device definition list. Refer to the appropriate language reference manual for information on accessing these fields.

Obtaining Information about a Particular Program Device

You can obtain information about a program device in two ways:

- Using the program device definition list
- Using the get-attributes operation

Program Device Definition List

After a program device is acquired, the program device becomes part of the program device definition list. The program device definition list contains information about the program device, such as the program device class, device type, and invite state.

Get-Attributes Operation

The get-attributes operation can be used at any time after a file has been opened to determine the status of a particular program device. The program device does not need to be acquired. The operation gets the current status about the session in which your program is communicating. If the program device is not acquired, the information is obtained from the program device entry defined with the ADDICFDEVE or OVRICFDEVE command.

The status information received by the get-attributes operation contains the fields shown in Figure 5-2.

Figure 5-2 (Page 1 of 2). Attribute Information Fields

Position	Value	Meaning
1 through 10	Name	Program device name: The name the program used to identify the program device in the file to read and write from.
11 through 20	Name	Device description name: The device description associated with the program device name (specified during configuration and optionally on the ADDICFDEVE or OVRICFDEVE command).
21 through 30	Name	User ID: If the program was started locally, this is the user ID used to sign on to the work station. If the program was started as a result of a program start request, this is the user ID used to start the target program.
31	I D U	The device is an ICF device type. The device is a display device. Unknown.
32 through 37	APPC SNUF BSCCL ASYN INTRA FNC RETAIL	APPC communications type. SNUF communications type. BSCCL communications type. Asynchronous communications type. Intrasystem communications type. Finance communications type. Retail communications type.
38	Y N	This is a requesting program device. This is a program device acquired by a source program.
39	Y N	Program device has been acquired. Program device has not been acquired.

Figure 5-2 (Page 2 of 2). Attribute Information Fields

Position	Value	Meaning
40	Y	Input is invited for this program device.
	N	Input is not invited for this program device.
41	Y	Invited input is available for this program device.
	N	Invited input is not available for this program device.
42 through 50	Reserved	Not applicable to communications.
51	Y	Session has an active transaction.
	N	Session does not have an active transaction.
52 ¹	0	Synchronization level is NONE.
	1	Synchronization level is CONFIRM.
53 ¹	M	Mapped conversation.
	B	Basic conversation.
54 through 61	Name	Remote location name: This is the remote location associated with the program device name (specified during configuration and on the ADDICFDEVE or OVRICFDEVE command).
62 through 69 ¹	Name	Local logical unit (LU) name.
70 through 77 ¹	Name	Local network ID.
78 through 85 ¹	Name	Remote LU name.
86 through 93 ¹	Name	Remote network ID.
94 through 101 ¹	Name	Mode: This is the mode associated with the program device name (specified during configuration and optionally on the ADDICFDEVE or OVRICFDEVE command).
102 through 144 ¹	Reserved	
145 through 146 ¹	Binary	ISDN remote number length in bytes, including type and plan.
147 through 148 ¹	00	ISDN unknown remote number type.
	01	ISDN international remote number type.
	02	ISDN national remote number type.
	03	ISDN network specific remote number type.
	04	ISDN subscriber remote number type.
	06	ISDN abbreviated remote number type.
149 through 150 ¹	00	ISDN unknown remote number plan.
	01	ISDN/telephony remote number plan.
	03	ISDN data remote number plan.
	04	ISDN telex remote number plan.
	08	ISDN national standard remote number plan.
	09	ISDN private remote number plan.
151 through 190 ¹	Character	ISDN remote number (blank-padded EBCDIC).
191 through 194 ¹	Reserved	
195 through 196 ¹	Binary	ISDN remote subaddress length in bytes, including type.
197 through 198 ¹	00	ISDN NSAP remote subaddress type.
	02	ISDN user defined remote subaddress type.
199 through 238 ¹	Character	ISDN remote subaddress (0 padded hexadecimal).
239 ¹	Reserved	
240	0	Incoming ISDN call.
	1	Outgoing ISDN call.
	2	Non-ISDN connection.

¹ This information is valid only for some of the communications types. These fields will be blank if the information does not pertain to the communications type you are using.

Sending and Receiving Data

Data is sent between systems by using output (or write) and input (or read) operations. The read and write operations are done using a record format. The results of read and write operations are communicated to the program with ICF messages, major/minor return codes, high-level language status values, and an **I/O feedback area**.

The I/O feedback area is updated for every read/write operation. The I/O feedback area consists of two sections:

- The **common I/O feedback area** contains information relevant to all communications types.
- The **file-dependent I/O feedback area** contains information that can apply to one or more of the communications types.

Common I/O Feedback Area

The common I/O feedback area contains information in the following fields:

- **Output operation count.** A count of the number of successful output operations. This count is updated only when an output operation completes successfully.
- **Input operation count.** A count of the number of successful input operations. This count is updated only when an input operation completes successfully and data is received.
- **Output then input operation count.** A count of the number of successful output then input operations.
- **Count of other operations.** A count of the number of successful operations other than output and input operations (such as acquire and release operations).
- **Current operation.** A hex value representing the current (last requested) operation, sent as follows:

Hex 01	Input
Hex 05	Output
Hex 06	Output then Input
Hex 11	Release

Hex 12 Acquire

- **Record format name.** Name of the record format just processed. The record format is either specified on the I/O request or determined by the specified format selection processing option.
- **Device class and type.** A hex code representing a device class for ICF and the communications type used as follows:

Hex 0B0E	APPC
Hex 0B20	SNUF
Hex 0B0A	BSCCL
Hex 0B1F	Asynchronous
Hex 0B1E	Intrasystem
Hex 0B42	Finance
Hex 0B43	Retail
- **Program device name.** The program device name to which the last operation was issued.
- **Record length.** The record length of the last I/O operation based on the record format processed, not including any indicators or program-to-system fields (P-data fields).
- **Blocked record count.** The number of records sent or received on an I/O operation. For ICF, the value is always 1.
- **Record length.** The record length of the last I/O operation based on the record format processed, including indicators and P-data.

File-Dependent I/O Feedback Area

The file-dependent I/O feedback area contains information in the following fields:

- **Actual record length.** On input, this is the actual length of user data received from the remote system or device. When the data received is longer than the data requested (all the received data cannot be contained in the record format used), the length of data is provided, if known. If the actual length cannot be determined, the field is set to hex FFFFFFFF. When a partial record is received (the remainder of the record is never sent), the length of the data received is provided. If the input operation completes with an error (other than partial record or truncated record), the contents of the field are undetermined.

On output, the actual length is the length of data moved from the user's buffer to the output buffer to send to the remote system. If the output operation completes with an error, the contents of the field are undetermined.

- **ICF major/minor return code.** A 4-character code (2 characters representing the major code, 2 characters representing the minor code) indicating the results of each operation.
- **Negative response error data.** For some return codes, this field contains more detailed information about the reason for the error. Refer to the *APPC Programmer's Guide*, the *Finance Communications Programmer's Guide*, the *SNA Upline Facility Programmer's Guide*, and the *Retail Communications Programmer's Guide* for more information on this field.
- **Request-to-write indication.** This indication tells you if the remote system requested that the application program stop sending data and give permission (by issuing a read or an allow-write request) to the remote system to begin sending.
- **Remote format name.** The remote format name received from the program device on an input operation. This is valid when the FMTSLT option on the ADDICFDEVE or OVRICFDEVE command is *RMTFMT. See Chapter 6 for more information on the FMTNAME DDS keyword.
- **Mode.** Mode associated with the program device. Mode is for APPC only. Refer to the *APPC Programmer's Guide* for more information on modes.
- **Safe indicator.** This field shows that an end-of-text (ETX) control character has been received in the buffer, and it is only valid for BSC. The safe indicator is not set if BLOCK(*USER) was specified on the ADDICFDEVE or OVRICFDEVE command. Refer to the *BSC Equivalence Link Programmer's Guide* for more information on this indicator.

Refer to Appendix C for a summary of the fields and the communications types to which the information applies. Refer to the appropriate communications programming manual for specific details on pertinent fields.

Like the open feedback area, the support provided by the high-level language you use determines whether you can access this information.

Checking Return Codes

After each operation, an ICF return code is returned to your program. Your program checks this return code to determine:

- The status of the operation just completed
- The operation that should be done next

It is recommended that your program check these return codes at the completion of every operation to ensure that the operation completed successfully or that the appropriate recovery action is taken.

A summary of these return codes is described in Appendix B. These codes, or groups of codes, are also converted to language return codes. For example, the ICF codes are converted to RPG *STATUS values or to COBOL/400 file status values. C/400 programming language does not have file status values. These values are shown in a chart in the appropriate language chapter of this manual.

Each ICF return code consists of a 2-digit **major code** and a 2-digit **minor code**. The major code identifies the general condition for a group of return codes, and is usually sufficient to determine the action to be taken. The minor code identifies the specific condition and may indicate the specific action that should be taken next. For example:

```
8233
| |
| Minor code
|
Major code
```

In this example the major code **82** indicates that an acquire or open operation was not successful. The minor code **33** indicates that the operation failed because an ADDICFDEVE command or OVRICFDEVE command was not issued for the program device you are trying to acquire.

Usually, your program determines the action to take by checking only the major code or the language status code. However, you may need to

check minor codes for specific conditions that occur for your particular application or communications configuration.

For more information about major and minor return codes, refer to Chapter 8.

Writing to a Program Device

Use a write operation to send data to the remote system.

The ICF file supports a set of DDS processing keywords and system-supplied formats, used in conjunction with the write operation, to perform various communications functions. Refer to Chapter 6 and Chapter 7 for more information about specifying the communications function to be used with the write operation.

Note: Data can be written to only one program device for each write operation.

Figure 5-3 shows the use of the write operation when sending data.

- 1** Your program uses a write operation to send data to the remote system.

- 2** Your program receives a return code indicating the completion status of the operation.

- 3** If a successful return code is received, your program continues sending several records.

Inviting a Program Device

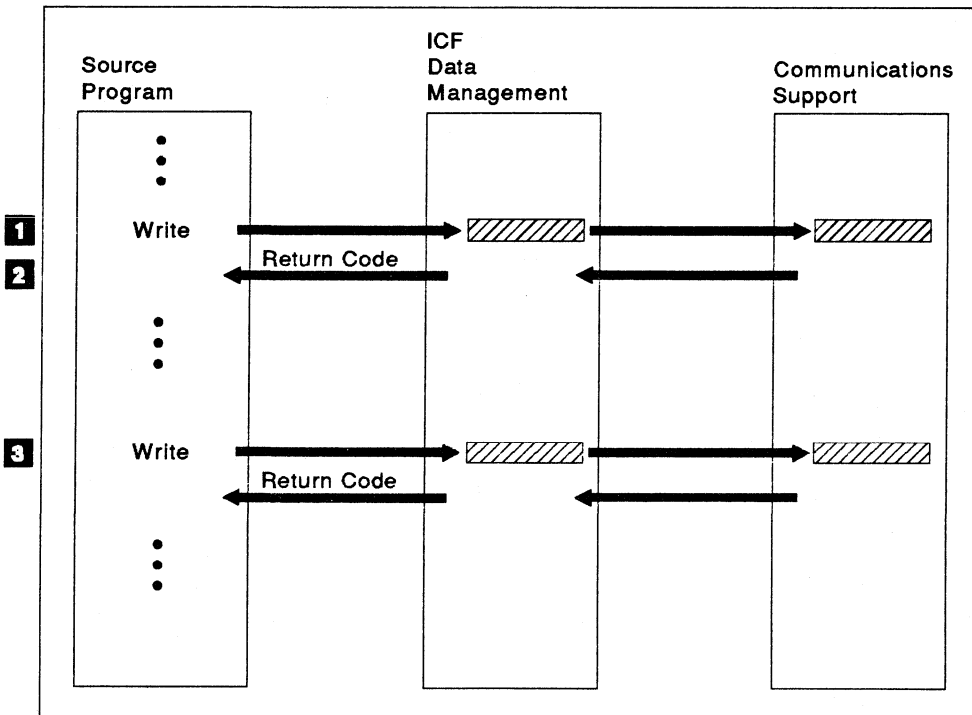
Your program indicates it wants to start an asynchronous input operation by inviting a program device. The invite function prepares your program to receive data. Your program can continue processing after issuing the invite request, and does not have to wait for the data.

An invite function is specified by:

- Issuing a write operation to a program device using a record format with the INVITE DDS keyword in effect.
- Issuing a write operation to a program device using a system-supplied format whose definition contains the invite function.

For more information about specifying the invite function, see Chapter 6 and Chapter 7.

AS/400 System



RSL5660-3

Figure 5-3. Using Write Operation When Sending Data

You can receive the response from an invited program device by doing an input operation. See “Reading from Invited Program Devices” on page 5-11 and “Reading from One Program Device” on page 5-16 for more information.

At least one program device must be invited before a read-from-invited-program-devices operation can return a response from a program device.

A program would invite a program device because:

- Inviting several program devices allows the program to do a read-from-invited-program-devices operation and receive data from one of the invited program devices with a response available. Therefore, using the invite function of the ICF file, the program can handle receiving a record from any one of several invited program devices by reading from a single point in the program. The program issues an input operation for the response.
- A program wants to use the time-out capability of the read-from-invited-program-devices operation.

Format Selection Processing

The format selection (FMTSLT) parameter on the ADDICFDEVE and OVRICFDEVE commands determines how ICF data management selects the record format to use when receiving data with the read and read-from-invited-program-devices operations. The three different methods of selecting a record format are discussed here. The name of the record format selected is placed in the I/O feedback area. Refer to “Determining the Record Format Returned” on page 5-16 for more information about determining the record format selected.

Program Selection (*PGM): If you specify FMTSLT(*PGM), which is the default, on the ADDICFDEVE or OVRICFDEVE command, your program must specify the record format to use when your program does an input operation. If no record format name is given, ICF data management uses the default record format. The default format is always the first format defined in the file.

The only selection process that is applicable when using the system-supplied QICDMF file is FMTSLT(*PGM). The default record format in this file is a 4096-byte data record called DFTRCD. You should either specify this format on the input operation or allow the system to default. Your program must then examine the input data to determine what data processing to perform on the fields in the record.

Record-Identifier Selection (*RECID):

Selecting FMTSLT(*RECID) on the ADDICFDEVE or OVRICFDEVE command provides a means of identifying and selecting the record format to use based on the data received. If you specify FMTSLT(*RECID), the file is searched for the RECID keyword on each input operation. The RECID keyword provides a definition for determining which record format to use.

When you specify the RECID keyword, you define a compare value. You must define the beginning position in the record format and the compare value to use. When data is received, the corresponding positions in the record are compared to the defined RECID values. When a match is found, that record format is used to process the received data. If no match is found, or if no data is received, the default record format is used.

When the FMTSLT(*RECID) is specified, the default format for an ICF file is one of the following:

- The first format in the file without the RECID keyword specified
- The first format in the file if all formats have the RECID keyword specified (applies only when no data is received)

Notes:

1. An ICF return code of 81E9 is returned to your program if the default format has the RECID keyword specified and no match is found for the received data. Refer to Appendix B for a complete list of return codes.
2. If a read with a record format specified is issued, the format specified must match the name determined by the RECID keyword selection process. If not, return code 3441 is returned to your program.

Refer to Chapter 6 for more information about the RECID keyword.

Remote Format Selection (*RMTFMT):

Remote format selection is supported by APPC and intrasystem communications.

If you specify FMTSLT(*RMTFMT) on the ADDICFDEVE or OVRICFDEVE command, your program does not need to enter a format name when it does an input operation. Instead, the format name passed with the data from the remote program is used. If the remote system is an AS/400 system, the remote program must specify the FMTNAME keyword in the record used to send the data to ensure the format name is sent.

If the remote system does not send a format name (for example, a record is sent without a FMTNAME keyword specified) and a format name is specified with the input operation in your program, that name is used to process the data received. If no format name is specified on the input operation, the default record format in the ICF file is used. The default record format is the first record format in the file.

If the remote program sends a format name and your program specifies a format name, the names must match. If they do not match, return code 3441 is returned to your program.

If the remote program sends a format name and FMTSLT(*RMTFMT) was not specified on the ADDICFDEVE or OVRICFDEVE commands, the remote format name sent is ignored by ICF.

The record format name received from the remote system on a successful input operation is put in the file-dependent section of the I/O feedback area. The high-level language may access this area to determine the remote record format name received from the remote system. Refer to the appropriate language reference manual for more information about accessing the I/O feedback area.

Summary of Format Selection Pro-

cessing: Figure 5-4 on page 5-11 summarizes the record format that is selected based on the format selection option specified on the FMTSLT parameter and the record format name specified on the input operation (if one was specified). This chart also shows what return codes can result from the format selection process on an input operation.

Figure 5-4. Format Selection Options

FMTSLT Option	Input Data	Record Format Name Specified on Input Operation	Record Format Name Not Specified on Input Operation
*PGM	Does not apply to format selection.	If specified format name is defined in ICF file, specified format selected. Otherwise, return code 83E0 returned to program.	Default format ¹ selected.
*RECID	Data received matches record format in file with RECID keyword.	If matched format is same as specified format, matched format selected. Otherwise, return code 3441 returned to program.	Matched format selected.
	Data received does not match any record in file with RECID keyword.	If default format ² does not have RECID keyword, default format selected. Otherwise, return code 81E9 returned to program. If format selected (default ²) is not same as specified format, error return code 3441 returned to program.	If default format ² does not have RECID keyword, default format selected. Otherwise, return code 81E9 returned to program.
	No data received.	If default format ² is same as specified format, default format selected. Otherwise, return code 3441 returned to program.	Default format ² selected.
*RMTFMT	Record format name (remote format) received from the remote system	If remote format name is defined in the ICF file, remote format selected. Otherwise, default ¹ format selected. If format selected is not same as specified format, return code 3441 returned to program.	If remote format is defined in the ICF file, remote format selected. Otherwise, default format ¹ selected.
	Record format name (remote format) not received from the remote system	If specified format name is defined in the ICF file, specified format selected. Otherwise, error return code 83E0 returned to program.	Default format ¹ selected.
<p>¹ The default record format for FMTSLT(*PGM) and FMTSLT(*RMTFMT) is the first record format in the ICF file.</p> <p>² The default format for FMTSLT(*RECID) is the first record format in the ICF file that does not have a *RECID keyword specified, or the first record format if all record formats have the *RECID keyword specified.</p>			

Reading from Invited Program Devices

The primary purpose of the read-from-invited-program-devices operation is to provide a single point in the program at which the program can wait for and receive a record from one of several program devices. The read-from-invited-program-devices operation can wait for and return a record to the program from one of the

invited program devices with an available record.

The read-from-invited-program-devices operation is also used to check whether the timer that was set by the timer function has ended. Refer to Chapter 6 and Chapter 7 for more information on the timer function.

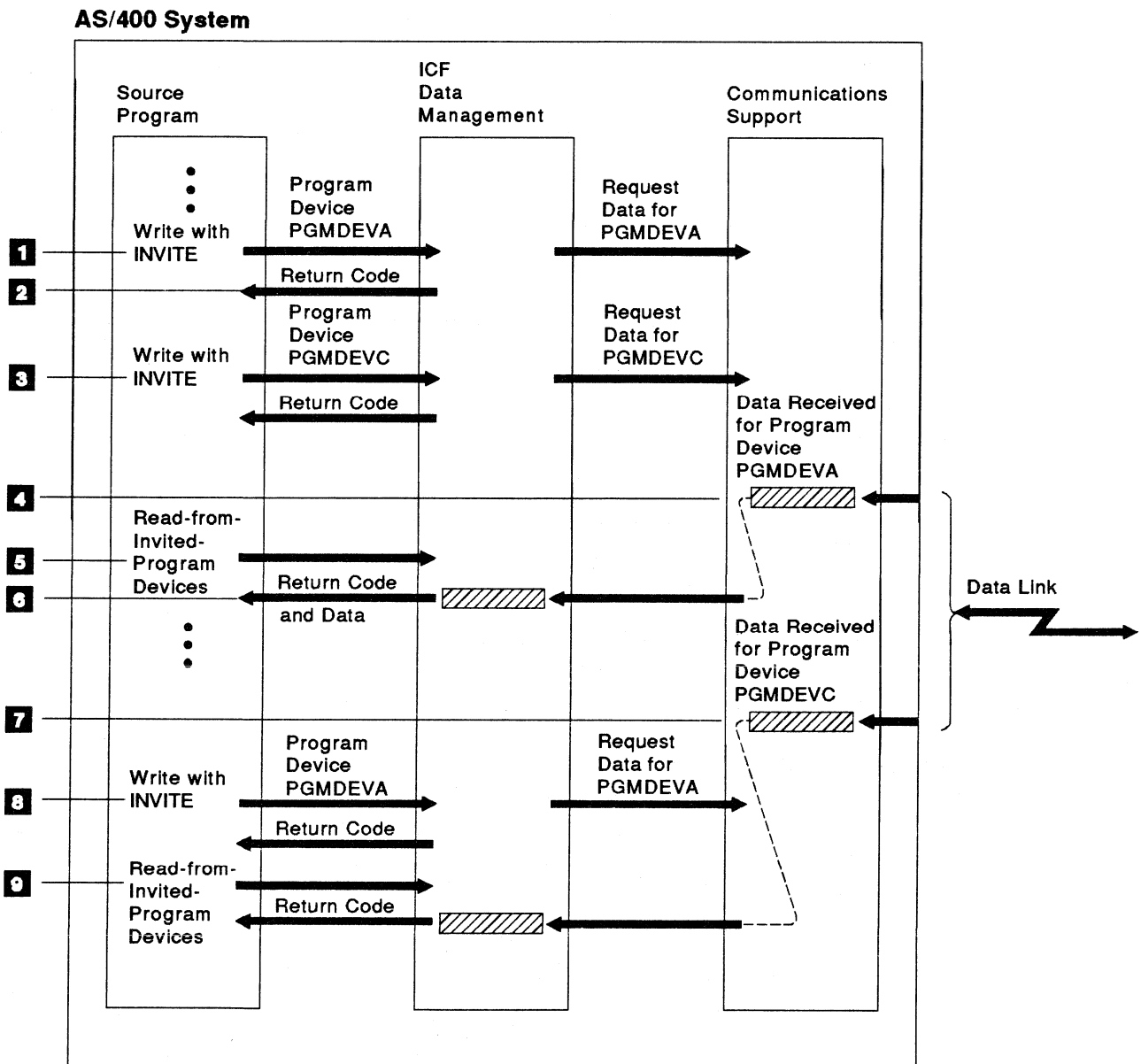
The read-from-invited-program-devices operation can complete when:

- A complete record arrives for an invited program device.
- A communications failure is detected on one of the invited program devices.
- The job is being canceled (controlled).
- The time specified by either the timer function or the WAITRCD parameter on the CRTICFF command is reached.

using considers the ICF file to be a multiple device file. C/400, COBOL/400, and RPG/400 languages consider the ICF file to be a multiple device file, while the FORTRAN/400 language does not. See the appropriate language reference manual for more information.

Figure 5-5 shows how you can use the invite function and read-from-invited-program-devices operation to receive data from two different program devices.

The read-from-invited-program-devices operation is only valid if the high-level language you are



RSL5126-7

Figure 5-5. Using the Invite Function and Read-from-Invited-Program-Devices Operation to Receive Data

- 1** Your program uses the invite function to ask the remote system to send data for program device PGMDEVA.
- 2** A successful completion return code tells your program that ICF data management received the operation and is asking the remote system to send data. No data has yet been received.
- 3** Your program issues another invite function. This invite is for program device PGMDEVC. Data has not yet been received for the first invite for program device PGMDEVA.
- 4** The system receives data for program device PGMDEVA. (Data is not necessarily received in the order in which the invite functions were issued. For example, data can be received for program device PGMDEVC before data is received for PGMDEVA. Your program must check the program device name to determine for which program device the data is received.)
- 5** A read-from-invited-program-devices operation is used to receive the data sent.
- 6** This time a successful completion return code tells your program that data has been received and is in your program buffer.
- 7** Data is received for program device PGMDEVC.
- 8** Another invite function is used to ask for program device PGMDEVA data.

- 9** Another read-from-invited-program-devices operation is used to receive the data for program device PGMDEVC.

Specifying Maximum Wait Interval: You can specify the maximum amount of time your program will wait for a read-from-invited-program-devices operation to complete.

The time interval can be specified by:

- Specifying the WAITRCD parameter on the CRTICFF, CHGICFF, and OVRICFF commands
- Issuing the timer function

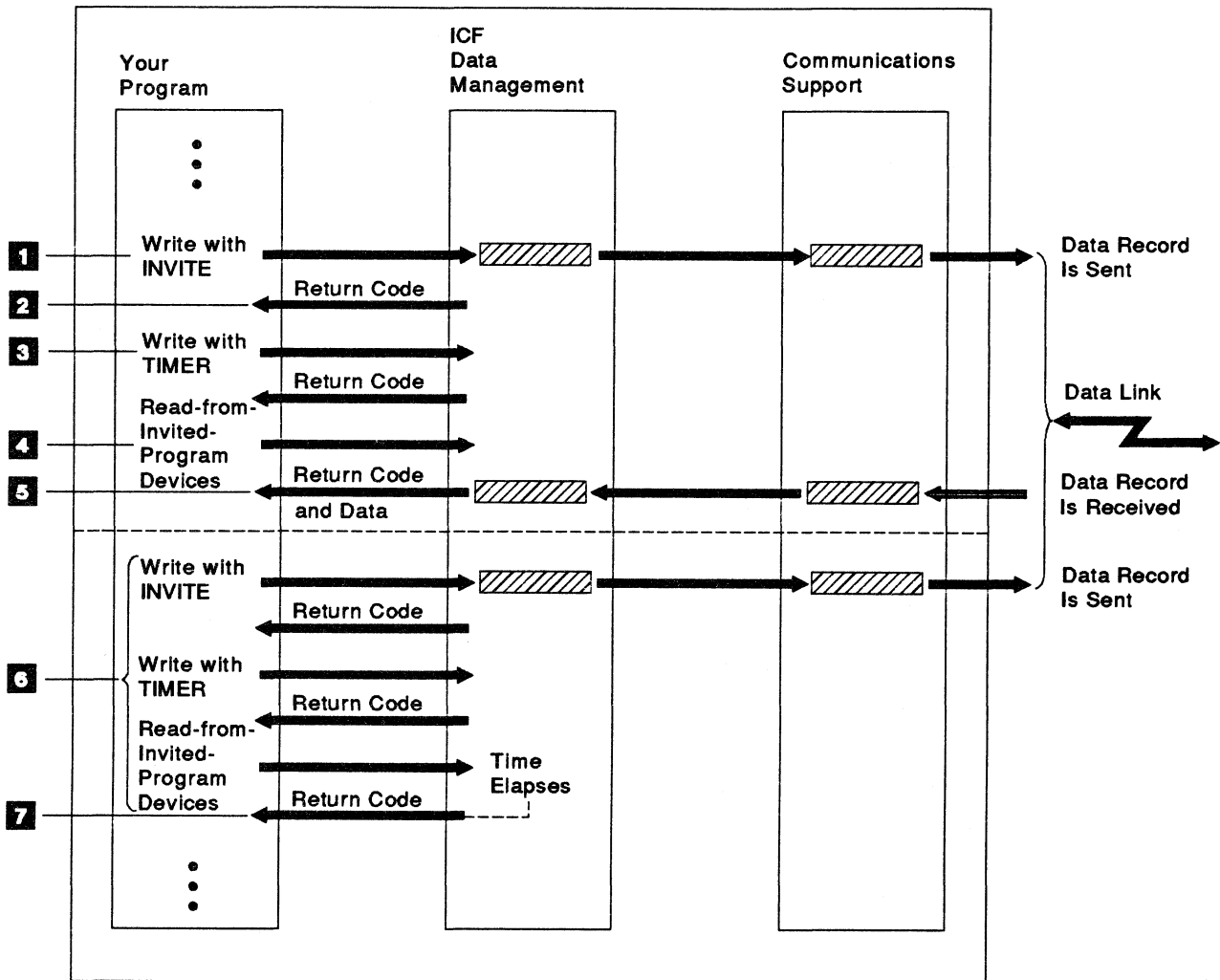
The WAITRCD parameter establishes the maximum time interval used for all read-from-invited-program-devices operations issued against the file.

The timer function is used to specify the maximum time interval used for read-from-invited-program-devices operations until either the timer ends or a new interval is set using the timer function. When the interval for the timer operation is in effect, the value specified on the WAITRCD parameter is ignored.

If a response is not received from the invited program devices within the specified amount of time, the program is notified with an ICF return code (0310) indicating that the timer interval has ended.

Figure 5-6 on page 5-14 shows the relationship between the timer function and the read-from-invited-program-devices operation.

AS/400 System



PSLS127-6

Figure 5-6. Relationship between Timer and Read-from-Invited-Program-Devices Operations

- 1** Your program issues an invite function.
 - 2** A successful completion return code tells your program that ICF data management accepted the request and that the remote system is expected to respond. No data has yet been received.
 - 3** Your program uses the timer function to set the maximum length of time to wait for a response.
 - 4** A read-from-invited-program-devices operation is used to read the data or, if the data is not received within the length of time specified, the return code indicates that the time you set has elapsed.
 - 5** The data is received before the time elapses. Therefore, a successful completion return code is passed to the program with the response from the remote system.
 - 6** Your program again uses the invite function to send data and ask for a response. The time interval is again set, and another read-from-invited-program-devices operation is issued.
 - 7** This time the read-from-invited-program-devices operation results in a return code that tells your program the time elapsed before a response was received from the remote system. When this happens, you may want to send a message to the operator, continue processing, or both.
- COBOL/400 programming language provides a means of calling the read-from-invited-program-devices operation as if the WAITRCD(*IMMED) is

specified. Refer to the *COBOL/400* Reference manual* for information about the NODATA phrase. See “Determining the Wait-for-Record Value” on page 4-6 for information about specifying the WAITRCD parameter.

Note: WAITRCD has no meaning to the FORTRAN/400 language, since the read-from-invited-program-devices operation is not supported.

Refer to Chapter 6 and Chapter 7 for information on specifying the timer function.

Responses: A program can receive one of many responses from the read-from-invited-program-devices operation. These responses are communicated to your program through an ICF return code.

The ICF file also supports a set of DDS response indicators that can be used in conjunction with the read-from-invited-program-devices operation to indicate status information about the operation. Refer to Chapter 6 for more information about response indicators.

The following sections describe possible responses from the read-from-invited-program-devices operation and the conditions under which the program receives the response.

Data from One of the Invited Program Devices with Data Available: If at least one invited program device has data available and the job has not been ended (controlled), the read-from-invited-program-devices operation returns data from one of the invited program devices.

After the read-from-invited-program-devices operation completes, the program can examine feedback that allows it to identify the program device that returned the data and the record format of the data returned. See “Determining Which Invited Program Device Had Data Available” on page 5-16 for information about how to determine which program device responded. See “Determining the Record Format Returned” on page 5-16 for information about determining the format of the returned record.

The program device returning the data is no longer in the invited state. The program device must be invited before it can return any more data using the read-from-invited-program-

devices operation. Other invited program devices remain invited.

Job Ended (Controlled): The read-from-invited-program-devices operation returns this response if the job is ended (controlled) before or during the wait for data to become available from an invited program device.

Receiving the job ended response does not cancel the invite. All invited program devices remain invited.

If any program in the job is notified that the job is being ended (controlled), that program should notify all other programs in the job. The system notifies only one program regardless of how many ICF files are used in the job.

When a program receives a job ended (controlled) indication, the program should complete operations and end before the system changes the job ended (controlled) to job ended (immediate) and forces all processing to stop. This action is important if a program needs to complete some processing before it ends.

No Invited Program Devices Have Data Available: The read-from-invited-program-devices operation returns this response when the job is not being ended and:

- At least one program device is invited.
- No data is available from any of the invited program devices.
- The WAITRCD(*IMMED) parameter is specified.

All invited program devices remain invited.

Time-Out on Wait for Data from Invited Program Devices: The read-from-invited-program-devices operation returns this response when:

- No data is available from any of the invited program devices in the amount of time specified as the WAITRCD parameter or the timer function.
- The job is not being ended.

All invited program devices remain invited.

No Program Devices Invited: If no program devices are in the invited state and the timer function is not in effect, the program is notified that no program devices were invited.

Error from One of the Invited Program Devices:

The read-from-invited-program-devices operation can return an error condition instead of data from one of the invited program devices if:

- The job is not being ended.
- At least one invited program device has a response available.

If an invited program device detects an error while it does the input operation, the error (like the data) is held until the program device is read using a read-from-invited-program-devices or read operation.

Determining Which Invited Program

Device Had Data Available: After a read-from-invited-program-devices operation returns data from an invited program device, the program may need to identify the name of the program device from which the data was returned. This identification is necessary if the program wants to handle one program device differently from other program devices.

The program can determine the name of the program device that returned the data from a field in the I/O feedback area. Refer to the appropriate language reference manual to learn about other ways to get this information, and about how to access the I/O feedback area.

If the program needs the name of the program device that returned the data, the program must get that information before doing any other I/O operations to the file.

If the read-from-invited-program-devices operation did not return a data available response (some other response, like job ended (controlled) was returned to the program), the field containing the name of the program device to which I/O was last directed in the I/O feedback area is set to *N (not applicable).

Determining the Record Format

Returned: Because a read-from-invited-program-devices operation returns a record from one of the invited program devices with an available record, regardless of that record's format, you cannot specify a record format on the read-from-invited-program-devices operation.

The system uses the FMTSLT parameter on the ADDICFDEVE, CHGICFDEVE, and OVRICFDEVE

commands to determine the record format.

Therefore, the program may have to determine the format of the record returned before handling the record (if a program device can return a record in one of several record formats).

The program can determine the record format of the data returned from a field in the I/O feedback area that indicates the name of the last record format used for I/O. Refer to the appropriate language reference manual to learn about other ways of getting this information and about how to access the I/O feedback area.

Note that if the program needs the name of the record format used to receive the data, the program must get that information before doing any other I/O operations to the file.

Reading from One Program Device

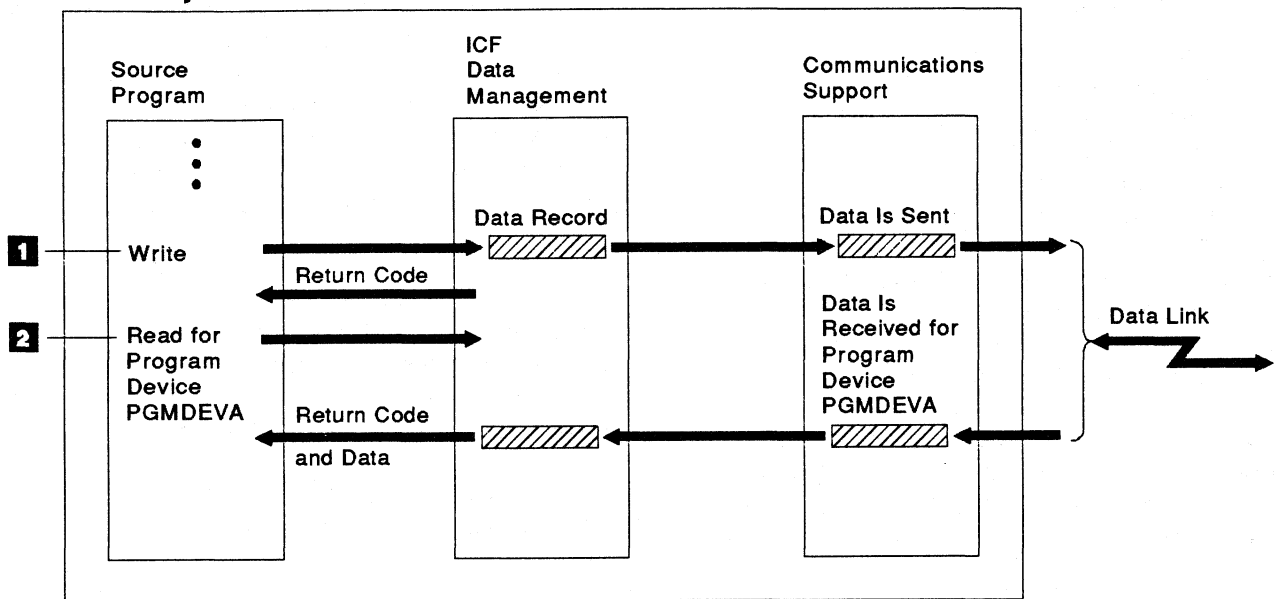
A read operation waits for and receives data from one program device. There is no time limit on a read operation (the WAITRCD parameter and the interval specified on a timer function are ignored). The program waits until data is available from that program device. The read operation differs from the read-from-invited-program-devices operation in that a read operation is directed to a specific program device, whereas the read-from-invited-program-devices operation receives data from any program device that was previously invited.

The program can indicate to the system that a read operation must be done in three ways:

- Explicitly, by specifying the name of a program device on the read operation
- Implicitly, by specifying the name of a record format on the read operation
- Implicitly, by specifying neither a record format or program device name, and the high-level language considers the ICF file a single device file

Because you cannot specify a record format on a read-from-invited-program-devices operation, the system interprets a read with a record format specified as a read operation. See the appropriate language reference manual for information about calling the read operation explicitly or implicitly and about which program device is used if the read operation is implicitly called.

AS/400 System



RSL5140-8

Figure 5-7. Using the Read Operation

Figure 5-7 shows how to use the read operation.

- 1** Your program uses a write operation to send data to the remote system.
- 2** A read operation is then issued to receive data from PGMDEVA. The program waits to receive the data before continuing.

If the program device specified on the read operation has been invited, the invite is satisfied and the input operation started when the program device was invited is completed before control is returned to the program.

The ICF file supports a set of DDS response indicators that can be used in conjunction with the read operation to indicate status information about the operation. Refer to Chapter 6 for more information about response indicators.

Writing and Then Reading from One Program Device

Some high-level languages support an interface to send a single I/O operation that does a write operation followed by a read operation to a program device.

The same record format is used on both the write and read operation.

Canceling an Invite of a Program Device

If a program device is invited, it is possible to cancel the invite:

- Explicitly, by issuing a cancel-invite function to the program device
- Implicitly, by issuing a write operation to the program device

Refer to Chapter 6 and Chapter 7 for information about specifying the cancel-invite function.

Waiting for a Display File, an ICF File, and a Data Queue

You can use data queues for a program that waits for data on a display file, an ICF file, and a data queue at the same time (in any combination). When you specify the DTAQ parameter for certain commands, you can indicate a data queue that will have entries placed on it when any of the following occurs:

- An enabled command key or the Enter key is pressed from an invited display device.
- Data becomes available from an invited ICF session.
- A user-defined entry is made to the data queue by a job running on the system.

The commands that allow you to indicate a data queue with the DTAQ parameter are:

- Create Display File (CRTDSPF)
- Change Display File (CHGDSPF)
- Override Display File (OVRDSPF)
- Create ICF File (CRTICFF)
- Change ICF File (CHGICFF)
- Override ICF File (OVRICFF)

By using the IBM-supplied QSNDDTAQ program, jobs running on the system can also place entries on the same data queue as the one specified in the DTAQ parameter.

For an ICF or display file, the application program uses the IBM-supplied QRCVDTAQ program to receive each entry placed on the data queue and then processes the entry based

on whether it was placed there by the display file, the ICF file, or the QSNDDTAQ program. For a display file, the application then issues a read or read-from-invited-devices operation to receive the data. For more information on the QRCVDTAQ function and syntax, and examples of waiting on one or more files and a data queue, see the *CL Programmer's Guide*.

The display file and ICF file entry that is put on the data queue is 80 characters in length and contains the field attributes described in Figure 5-8. Therefore, the data queue that is specified using the commands listed above must have a length of at least 80 characters.

Entries placed on the data queue by jobs using QSNDDTAQ are defined by the user.

Figure 5-8. Display File and ICF File Entry Field Attributes

Position	Data Type	Meaning
1 through 10	Character	The type of file that placed the entry on the data queue. This field can have one of two values: <ul style="list-style-type: none"> *ICFF (ICF file) *DSPF (display file) If the job receiving the data from the data queue has only one display file or one ICF file open, then this is the only field that needs to be used to determine what type of entry has been received from the data queue.
11 through 12	Binary	Unique identifier for the file. The value of the identifier is the same as the value in the open feedback area for the file. This field should be used by the program receiving the entry from the data queue only if more than one file with the same name is placing entries on the data queue.
13 through 22	Character	The name of the display or ICF file. This is the name of the file actually opened after all overrides have been processed, and is the same as the file name found in the open feedback area for the file. This field should be used by the program receiving the entry from the data queue only if more than one display file or ICF file is placing entries on the data queue.
23 through 32	Character	The library where the file is located. This is the name of the library after all overrides have been processed, and is the same as the library name found in the open feedback area for the file. This field should be used by the program receiving the entry from the data queue only if more than one display file or ICF file is placing entries on the data queue.
33 through 42	Character	The program device name after all overrides have been processed. This name is the same as that found in the program device definition list of the open feedback area. For file type *DSPF, this is the name of the display device where the command was entered or the Enter key was pressed. For file type *ICFF, this is the name of the program device where data is available. This field should be used by the program receiving the entry from the data queue only if the file that placed the entry on the data queue has more than one device or session invited prior to receiving the data queue entry.
43 through 80	Character	Reserved.

Releasing a Program Device

You can explicitly release a program device from an ICF file by using the release operation, or you can implicitly release the device by closing the file.

If you release the program device, you must reacquire it before you can use it again for I/O operations.

The release operation ends the session only when certain criteria are met. The end-of-session function always ends the session. Refer to Chapter 6 and Chapter 7 for more information about specifying an end-of-session function.

The following processing is done by the release operation:

- Source program
 - If the program device is invited, the release operation fails.
 - If a transaction is still active on the session, the release operation fails.
 - If a transaction is not active on the session, the session ends.
 - If the device description associated with the program device is allocated when the program device is acquired, it is deallocated when the program device is released.
 - If an error occurs due to the hardware or an SNA protocol violation, the release operation fails.
- Target program
 - The release operation severs the logical connection between the application and the requesting program device. The session is not ended.

- The program (or another program in the same job structure) can reestablish the connection to the same session by acquiring the requesting program device. The communications session, including the state of the session, remains intact.

Closing an Intersystem Communications Function File

The processing done by the close operation depends on whether the file is shared. If the file is not shared, the following processing is done:

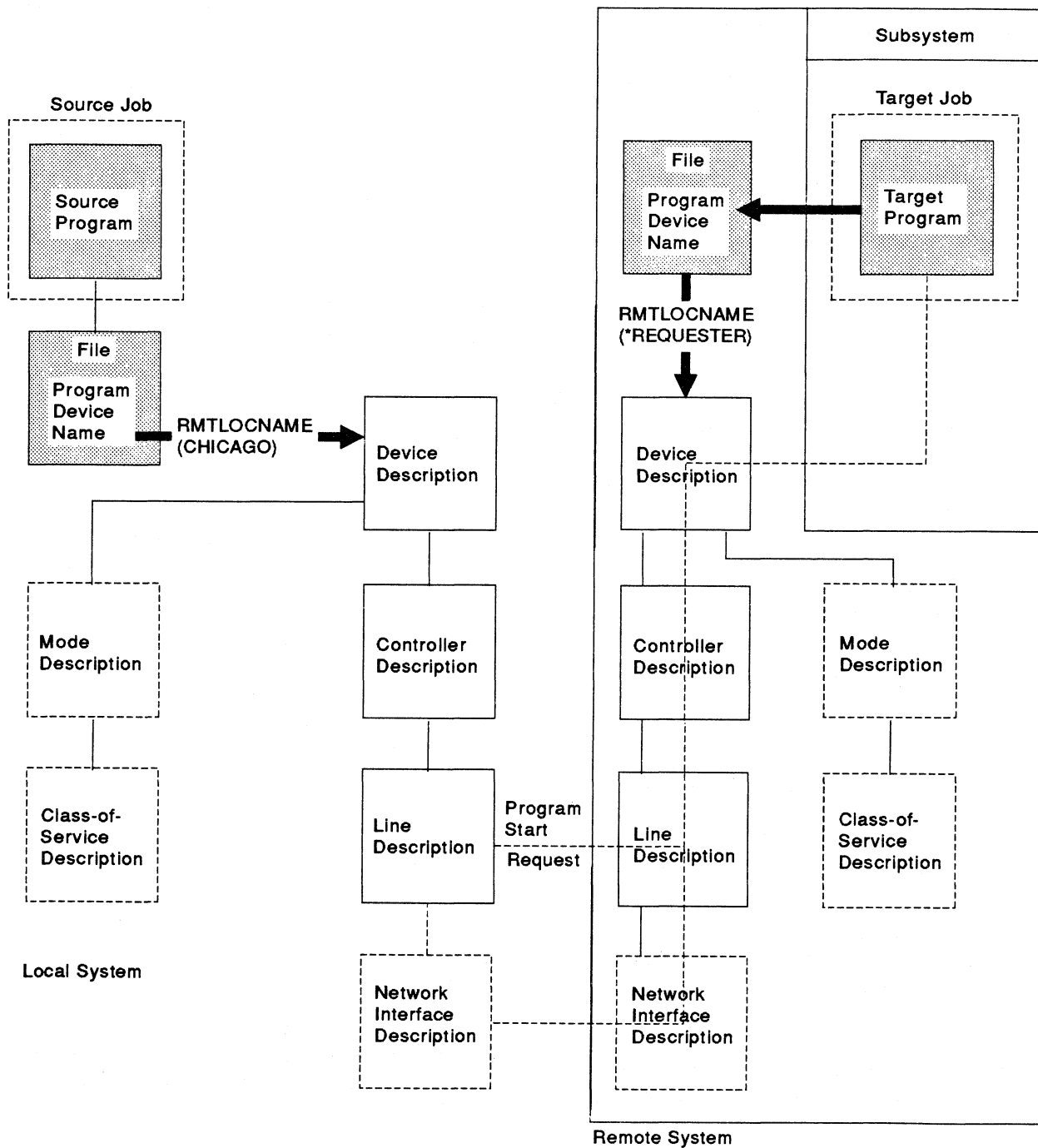
- All sessions associated with the source program are ended.
- All sessions associated with the target program are released.
- The file resources allocated by the open operation are deallocated and returned to the system.

If the file is shared, the program cannot do I/O operations to the file. Other programs that have the file open can still use the file.

If the close operation is successful, an open operation is the only program operation allowed to the file. If the close operation fails, the program should call the close operation a second time. A second close operation is always successful.

Summary

Figure 5-9 on page 5-20 shows the relationship between the program, the ICF file, and the communications configuration on a local and remote AS/400 system.



RSL5177-6

Figure 5-9. Relationship of Program, File, and Configuration

Notes:

1. The mode description and class-of-service description apply to APPC only.
2. Network interface description applies only to ISDN.

Local System

The source program is your C/400, COBOL/400, FORTRAN/400, or RPG/400 application, which is communicating with the target program through an ICF file. You can create this file using the CRTICFF command, and change it using the CHGICFF or OVRICFF command.

The source program:

- Opens the file
- Acquires one or more program devices (Only one is allowed per ICF file for FORTRAN/400 applications.)
- Reads and writes to program devices in the file to receive and send data
- Releases the acquired program devices
- Closes the file

The device entries defined in the file with the ADDICFDEVE, CHGICFDEVE, or OVRICFDEVE command provide:

- Mapping to the communications configurations
- Communications-type-dependent definitions of program device attributes

The local configurations selected by the program device entry define the connection to the remote system.

Remote System

An incoming program start request from the local system starts a target job.

The target program is your C/400, COBOL/400, FORTRAN/400, or RPG/400 application, which is communicating with the source program through

an ICF file. You can create this file using the CRTICFF command and change it using the CHGICFF or OVRICFF command.

The target program:

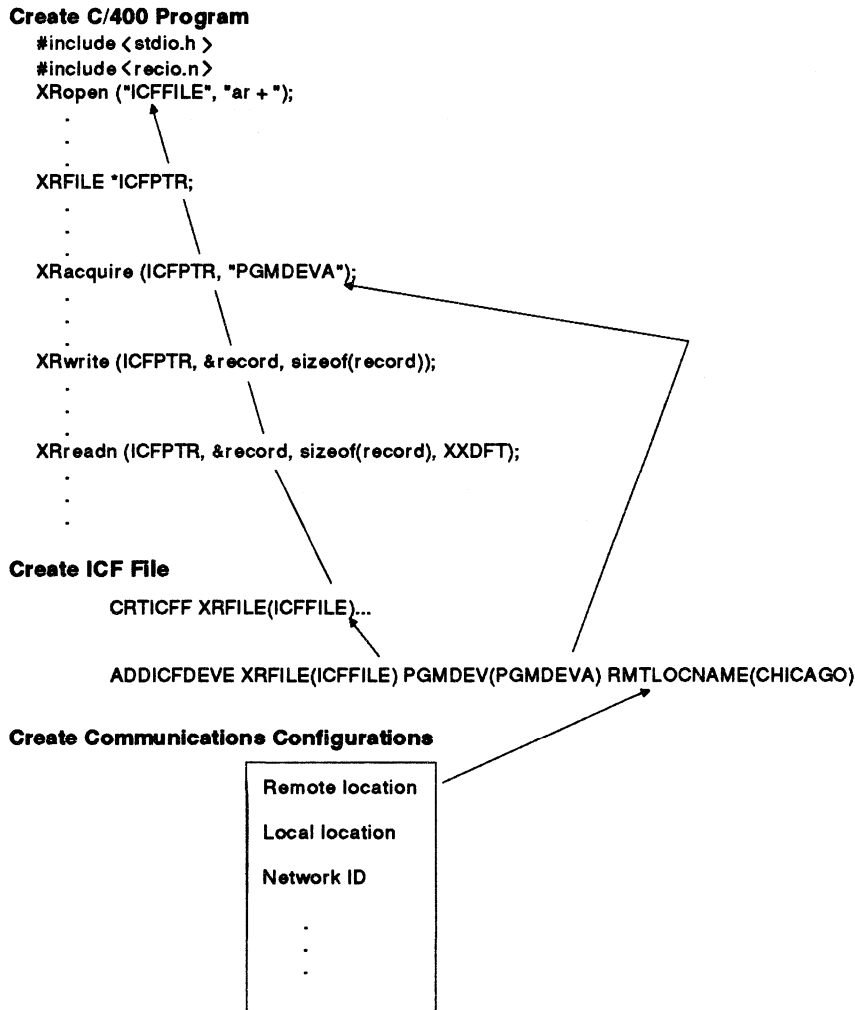
- Opens the file
- Acquires the requesting program device (This must be done implicitly on the open for FORTRAN/400 programs.)
- Reads and writes to the requesting program device for the file to receive and send data
- Releases the requesting program device (This must be done implicitly on the close for FORTRAN/400 programs.)
- Closes the file

The device entries defined in the file with the ADDICFDEVE, CHGICFDEVE, or OVRICFDEVE command provide:

- A relationship to the requesting program device
- Communications-type-dependent definitions of program device attributes

Refer to “Remote Program Start Considerations” on page 8-5 for more information on subsystems.

Figure 5-10 shows how the program, file, and configuration names are mapped to each other in C/400 programming language.



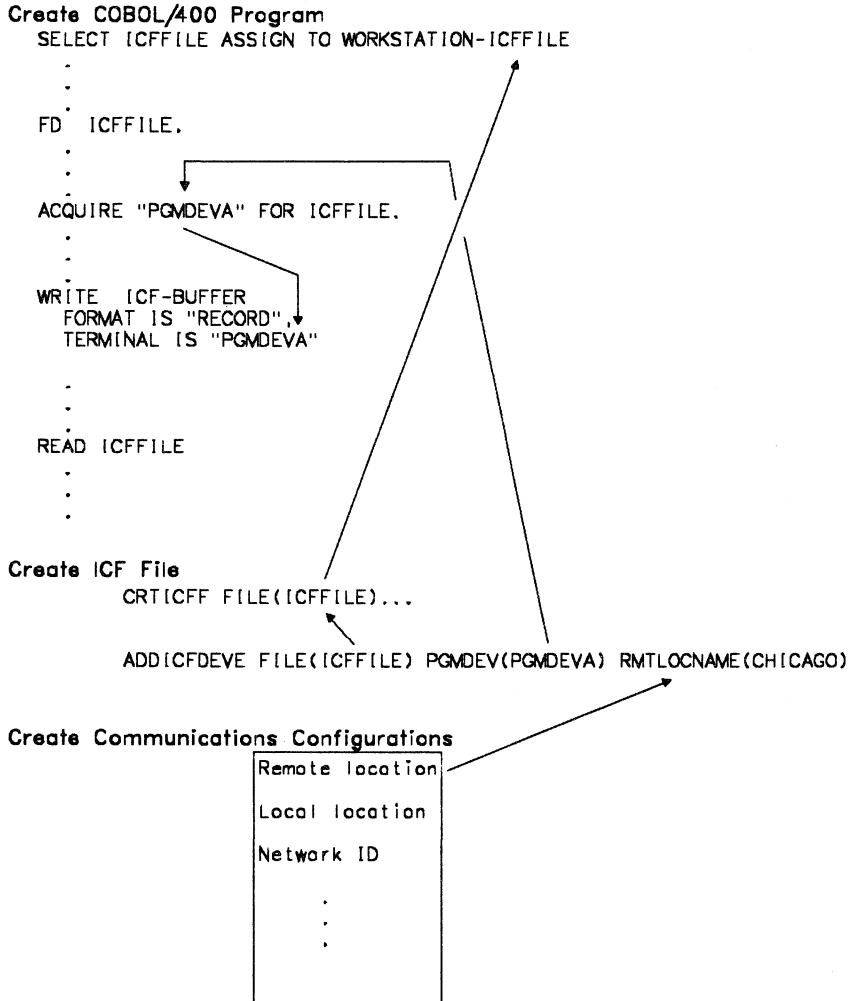
RSL5683-3

Figure 5-10. C/400 Program, File, and Configuration Mapping

You create various configuration objects when you use the communications configuration function. The remote location name provides the primary mapping between the program device and the communications configurations. The

specified remote location name is used to select the device description.

Figure 5-11 shows how the program, file, and configuration names are mapped to each other in COBOL/400 programming language.



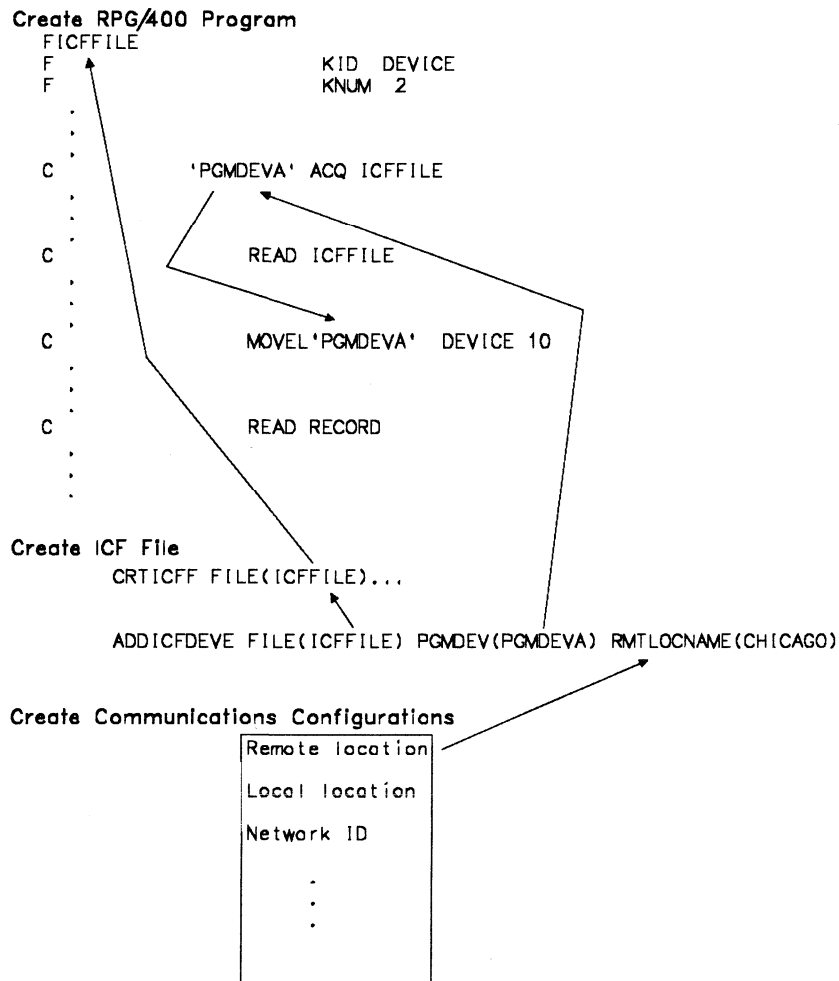
RSL5179-6

Figure 5-11. COBOL/400 Program, File, and Configuration Mapping

You create various configuration objects when you use the communications configuration function. The remote location name provides the primary mapping between the program device and the communications configurations. The specified remote location name is used to select the device description.

You create various configuration objects when you use the communications configuration function. The remote location name provides the primary mapping between the program device and the communications configurations. The specified remote location name is used to select the device description.

Figure 5-12 shows how the program, file and configuration names are mapped to each other in RPG/400 programming language.



RSL5178-7

Figure 5-12. RPG/400 Program, File, and Configuration Mapping

Chapter 6. Using Communications DDS Keywords

This chapter explains how to use data description specifications (DDS) keywords on input and output operations to perform communications functions with the remote system. These DDS keywords are associated with the defined record format used on the read or write operation. The record formats associated with the DDS source for your ICF file are referred to as user-defined formats. This is in contrast to the system-supplied formats discussed in Chapter 7. It is assumed that you have opened your file and established your session as described in Chapter 5.

The information and illustrations provided describe the function of each of the processing keywords supported by the ICF file. Although all of the parameters supported by each keyword are described, the information on coding the keywords is found in Chapter 6 of the *DDS Reference*. The *DDS Reference* also contains general information on defining record formats.

You can use several DDS keywords and combinations of keywords on a single input/output (I/O) operation. Figure 6-17 on page 6-24 shows the processing sequence when multiple DDS keywords are specified together.

All the keywords described in this chapter may not be supported by the communications type you are using. Furthermore, some keywords may operate differently depending on the communications type. Figure 6-15 on page 6-22 and Figure 6-16 on page 6-23 summarize the support provided by each communications type. Refer to the appropriate communications programming manual for the communications type you are using for more detail about supported keywords.

Several DDS keywords that do processing-control, referencing, and text-definition functions that are valid in ICF files and other types of files are not discussed in this manual. These keywords are ALIAS, FLTPCN, INDARA, INDTXT, REF, REFFLD, and TEXT. Refer to the *DDS Reference* for more information on how to code and use these keywords. These keywords are supported by all communications types.

Examples of source DDS and the commands used to create and use an ICF file are found at the end of this chapter.

Refer to Chapter 10 for complete program examples that use DDS keyword processing.

You can use system-supplied communications formats instead of DDS keywords to do communications-specific functions. Refer to Chapter 7 for more information on system-supplied formats.

Starting a Program on the Remote System

Your program must specify the target program it will communicate with before it can send or receive data. The target program is started by specifying an output operation with the EVOKE keyword in effect. Generally, the necessary parameters to identify the target program you want to start must be specified. However, for some communications types, these parameters are not required.

These parameters include items such as the program name, the name of the remote library where the program is stored, and security information (when required). You may also include data with the evoke function, which will be sent to the target program when the evoke function is done. When your program issues the evoke function, a program start request is sent to the remote system.

Use the EVOKE, SECURITY, and SYNLVL keywords to start a program at the remote system.

Evoke (EVOKE, SECURITY, and SYNLVL)

The EVOKE keyword allows your program to start a program on the remote system. EVOKE is valid only when the source program is not already communicating with the target program on the same transaction.

The format of the EVOKE keyword is:

```
EVOKE([library-name/]program-name [parameter-1...
      [parameter-255]])
```

The program-name parameter is required on the EVOKE DDS keyword to identify the program to be started on the remote system. However, some communications types do not require the program name. In these cases, blanks should be used for the program name instead. Refer to the appropriate communications manual to determine if the communications type you are using requires a program name on the EVOKE DDS keyword.

The optional library-name parameter specifies the library where the program is stored on the remote system. In general, it is best to specify the library separate from the program. If you specify the target program as a single literal, then it must be specified in the format required by the remote system or in the architected format. For example, if you are using APPC with another AS/400 system, the program name can be in the form *library/program* or *program.library*. If the remote system is an AS/400 system and a library name is not given, the library list for the subsystem that is handling the request on the remote system is searched for the program name. The library list for the subsystem consists of the values from the QSYSLIBL and QUSRLIBL system values at the time the subsystem was started.

In addition to passing the program-name and library-name to the remote system, you can also use the EVOKE DDS keyword to send up to 255 user-defined parameters to the remote system. (Some communications types do not support 255 parameters. Refer to the appropriate communications programming manual for any additional restrictions.) The target program defines the number and format of the parameters. If the remote system is another AS/400 system, the following apply:

- The parameters are passed to the program as if they were passed from a Call a Program (CALL) command.
- If the parameters contain embedded commas, the remote AS/400 system considers these to be multiple parameters rather than a single parameter.

Any transaction status information sent by the source program is received on the first read

operation of the target program. For example, if the target AS/400 system program is started from an AS/400 system with an evoke-with-invite function using advanced program-to-program communications (APPC), the first read operation on the target program completes with an 0300 (change direction received) major/minor return code.

You can use the SECURITY DDS keyword to include security information with the evoke request. The SECURITY keyword is only valid in conjunction with the EVOKE keyword. All security specifications must satisfy the requirements of the remote system.

The format of the SECURITY keyword is:

```
SECURITY(n reserved-word|'literal'|field-name-1
         [&field-name-1[.3.]])
```

The n parameter required by the remote system identifies the security subfield being described. The n parameter can be specified as:

- 1 for a profile ID
- 2 for a password
- 3 for a user ID

You can specify the following values for the security fields:

reserved-word. This value can be specified as one of the following:

- *USER. Specifies that the user's profile name on the local AS/400 system is used as the security field.
- *NONE. Specifies that the security field is not supplied.

'literal'. A literal value of up to 10 characters that contains the needed security information.

field-name (or &field-name). The name of a field in the record format that contains the needed security information. If you want to send blanks as the security field, you must specify this as a literal value or use a field name.

If you do not explicitly define the security values on the SECURITY keyword for an evoke request, no security values are sent.

Refer to Chapter 8 for information about remote program start considerations on the AS/400 system.

Use the SYNLV L DDS keyword to specify the level of synchronization supported on this transaction. The SYNLV L keyword is valid only in conjunction with the EVOKE keyword.

The format of the SYNLV L keyword is:

SYNLV L[*NONE|*CONFIRM]

You can specify the following optional values for the SYNLV L keyword:

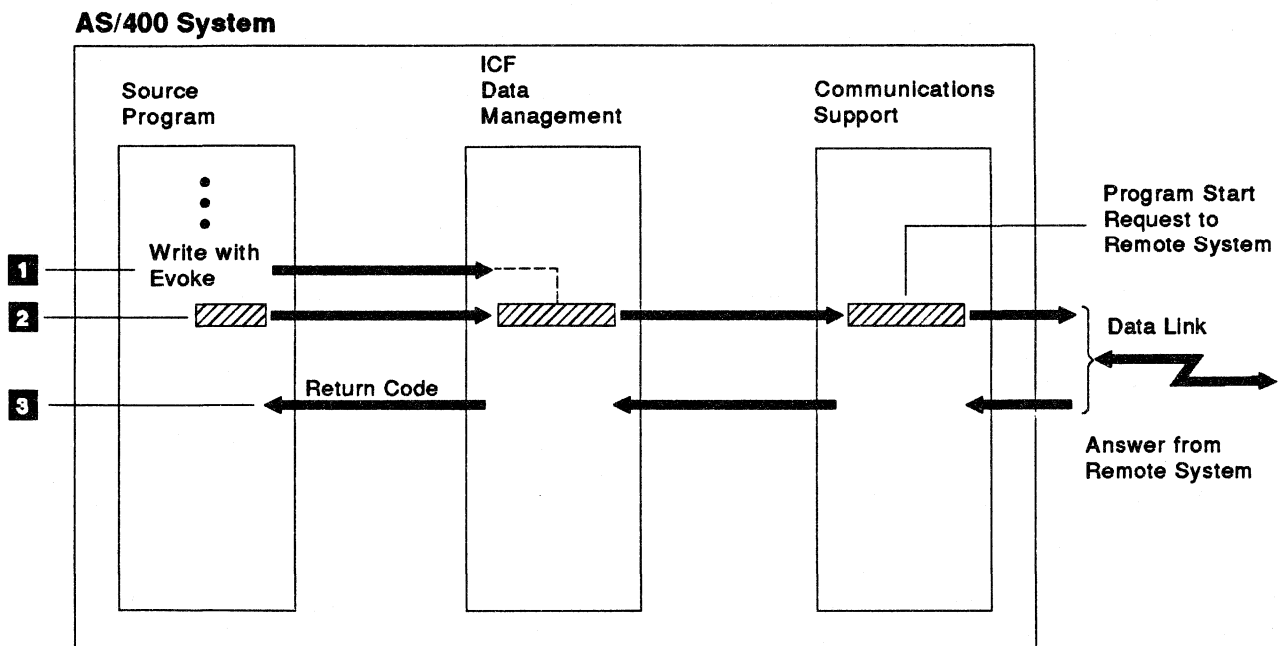
***NONE.** Specifies that confirmation of the receipt of data is not allowed on this transaction. For example, on the AS/400 system the CONFIRM keyword is not allowed with SYNLV L (*NONE).

***CONFIRM.** Specifies that the sending program can request that the receiving program responds to receipt of the data. The receiving program can send a positive response, or the receiving program or system can send a negative response. For example, on the AS/400 system the CONFIRM keyword is allowed on write operations.

Refer to the keyword descriptions for “Confirm (CONFIRM)” on page 6-4, “Receive-Confirm” on page 6-18, and “Respond-to-Confirm (RSPCONFIRM)” on page 6-12 for additional information on CONFIRM processing.

Figure 6-1 shows how to start a target program on the remote system.

- 1** The source program issues an evoke request to start the program at the remote system.
- 2** The evoke parameters, including program name, library name, and security information are sent to the remote system. Program initialization parameters can also be sent with the program start request (optional).
- 3** A successful completion return code tells the source program that the evoke request was accepted and a program start request was sent to the remote system. If the program start request is successful, both the program at the remote system and the communications transaction are started.



PSLS125-5

Figure 6-1. Starting a Target Program

Sending Data

You may begin sending and receiving data when both systems are communicating with each other. This section discusses sending data. See “Receiving Data” on page 6-7 for a discussion on receiving data.

You can use several DDS keywords and combinations of keywords in conjunction with sending data. These keywords provide additional information about how to process the data being sent to the remote program.

The following are valid functions that can be done when sending data. The DDS keywords associated with these functions are valid only with output operations.

Variable-Length Data (VARLEN)

The length of an output operation is determined by the record format specified. The record format length is determined by the record definition in DDS. You can use the VARLEN keyword to change the I1.variable length data record (VARLEN) keyword length of the data record sent with each write operation, while using the same record format.

The format of the VARLEN keyword is:

```
VARLEN(&field-name)
```

The field-name parameter specifies the length of the record sent on a write operation. The length cannot be greater than the length of the data field defined for this record format. The length you specify with the VARLEN keyword overrides any length specified elsewhere in your write operation.

Variable-Buffer-Management (VARBUFMGT)

Use the VARBUFMGT keyword to send or receive multiple or partial records, rather than just one record, with one record format per write or read operation.

Using the VARBUFMGT keyword allows you to specify the length of data independently of the data itself. A program uses the data length

specified as the value passed in the variable length (VARLEN) DDS keyword, or if VARLEN is not used, the length of the record format specified on the read or write operation. The length specified must be greater than zero.

This function is valid only for APPC. Refer to the *APPC Programmer's Guide* for more information.

Force-Data (FRCDTA)

Use the FRCDTA keyword on a write request to cause the communications support to immediately send any data currently held in the output buffer. The communications support does not wait for the buffer to fill. Any data specified on the same operation as the force-data request is also sent. No operation is done if there is no data in the buffer to send.

Note: This causes the data to be sent to the other system, but not necessarily to the remote program.

Confirm (CONFIRM)

Use the CONFIRM keyword to request that the remote program respond when it has received the data you sent. An output operation with the CONFIRM keyword specified forces any data in the output buffer to be sent. The CONFIRM keyword also asks the remote program to respond when the data is received. The operation does not complete, and your program does not continue, until a response is received. The remote program must respond with either a positive or negative reply as to whether the data was successfully received.

Note: Refer to the RCVCONFIRM keyword described in “Using Response Indicator Keywords” on page 6-18 and the RSPCONFIRM keyword described in “Additional Keywords” on page 6-12 for information on how to receive and respond to a confirm request.

If a positive response is received, the output operation completes normally. If a negative response is received, the major/minor return code and ICF message indicate the reason.

CONFIRM is valid only on a transaction with a synchronization level of confirm.

Format-Name (FMTNAME)

Use the FMTNAME keyword to pass the name of the record format used for this output operation to the remote system. If the remote system is an AS/400 system, ICF uses this name to find the record format to use when receiving the data at the remote system.

Note: If you use the FMTNAME keyword while sending data to another AS/400 system, you should specify *RMTFMT for the format selection (FMTSLT) parameter on the Add Intersystem Communications Function Device Entry (ADDICFDEVE), Change Intersystem Communications Function Device Entry (CHGICFDEVE), or Override Intersystem Communications Function Device Entry (OVRICFDEVE) command at the system at which the data is received.

Subdevice-Selection (SUBDEV)

Use the SUBDEV keyword to specify the remote system device (such as a printer or diskette) to which you are sending data. The receiving controller then directs output from your program to the appropriate device. The subdevice selection is designed primarily to support specific hardware devices, such as 3776, 3777, and 3780.

The format of the SUBDEV keyword is:

SUBDEV(type)

The type parameter values, *DC1, *DC2, *DC3, and *DC4, are required to specify the device control character used by the receiving controller so that output can be directed to the appropriate device.

End-of-Group (ENDGRP)

Use the ENDGRP keyword to indicate to the remote system the end of a user-defined group of records. The communications type you are using determines the type of indication sent to the remote system to indicate the end of a group of records.

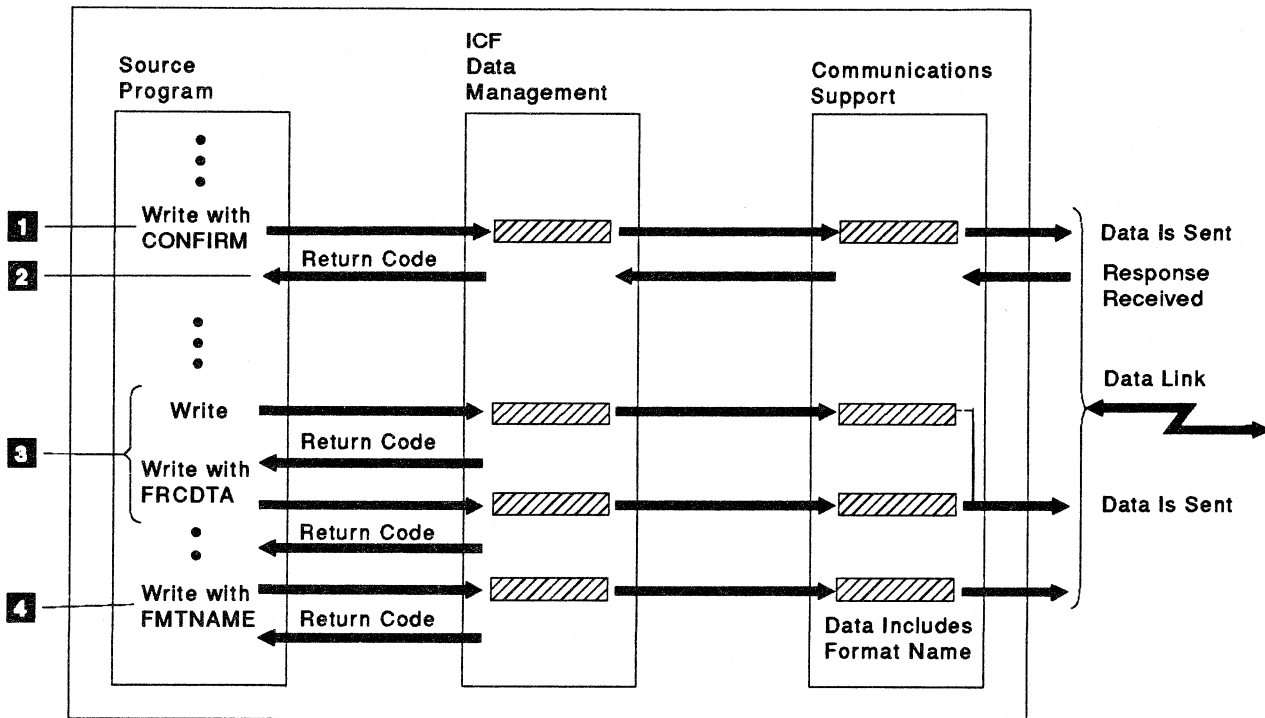
Note: Refer to the RCVENDGRP keyword described in “Using Response Indicator Keywords” on page 6-18 for information on how to handle receiving an end-of-group indication.

Function-Management-Header (FMH)

Use the function-management-header (FMH) keyword to send control information about the data that follows to the remote system. A function-management-header is valid only with the first record of a group.

Note: Refer to the RCVFMH keyword described in “Using Response Indicator Keywords” on page 6-18 for information on how to handle receiving a function-management-header.

AS/400 System



RSL5180-4

Figure 6-2. Using the CONFIRM, FRCDTA, and FMTNAME Keywords to Send Data

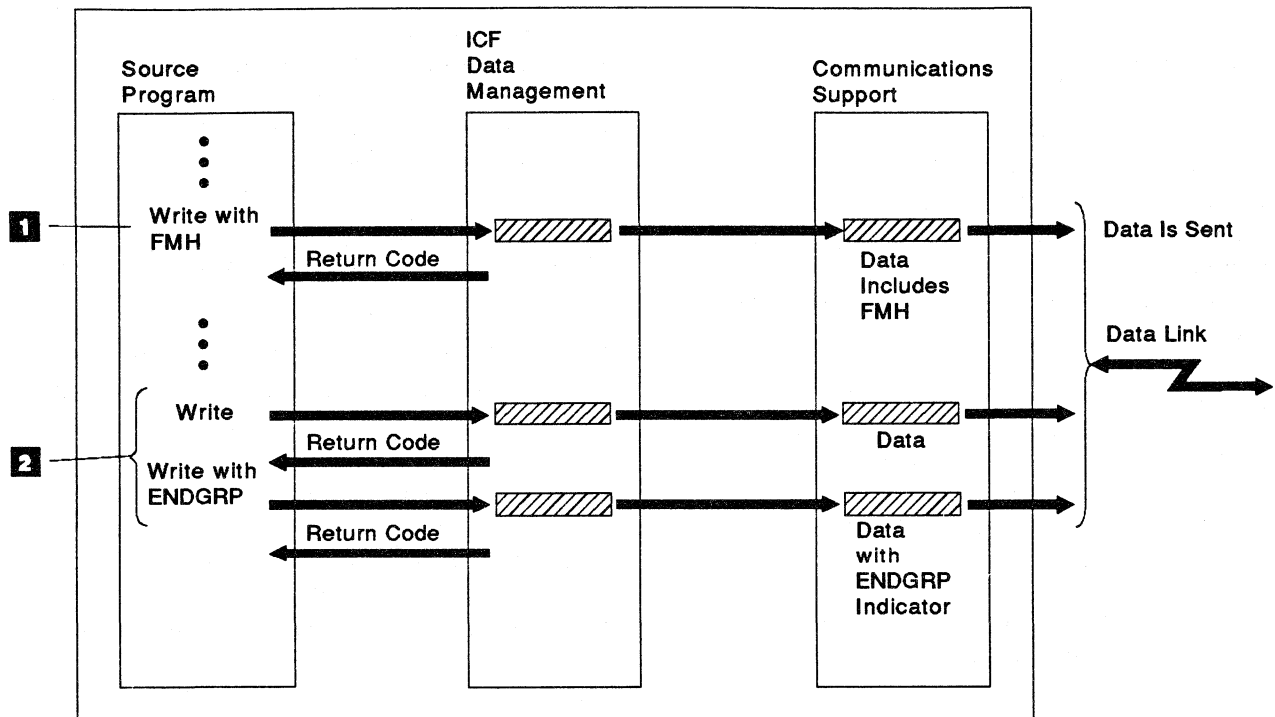
Examples of Sending Data

Figure 6-2 shows how to use the CONFIRM, FRCDTA, and FMTNAME keywords when sending data.

- 1** Your program issues a write-with-confirm operation to send data to the remote system and asks the remote system for a response.
- 2** Your program cannot continue processing until a response is received from the remote system. Your program checks the return code to determine if the remote system issued a positive or negative response.
- 3** If a successful return code is received, your program continues sending several records. On the last record, your program also specifies the force-data function. The FRCDTA causes all buffered data to be sent. The return code indicates the data is successfully sent. The force-data request does not wait for a response from the remote system.
- 4** Your program sends a record. The format-name function indicates that the record format name used on this write is also sent to the remote system. The remote system uses this record format when receiving the data.

Figure 6-3 on page 6-7 shows how to use the ENDGRP and FMH keywords when sending data.

AS/400 System



RSL5181-3

Figure 6-3. Using the ENDGRP and FMH Keywords to Send Data

- 1** Your program sends a record to the remote system and, with the FMH keyword, indicates that the first part of the data is function-management-header data, which contains information about the user data that follows.
- 2** Your program continues sending data records to the remote system. Your program uses the end-of-group function on the last record to indicate it is the last in this group of records.

Receiving Data

You can use two operations to receive data: read and read-from-invited-program-devices. In addition, you can use the invite, timer, and record-identification functions with the preceding operations to provide additional functions when receiving data.

The read operation receives data from the program device you specify. This operation differs from the read-from-invited-program-devices operation, which receives data from any

program device with a previously issued invite request.

Invite (INVITE)

The INVITE keyword prepares your program to receive data. You must do an output operation with the INVITE keyword specified to issue an invite function. You can combine additional output keywords or data with the invite function. Your program can continue processing after issuing the invite request, and does not need to wait for the data.

The read-from-invited-program-devices operation is a companion to the invite function. After issuing an invite function, you use the read-from-invited-program-devices operation to receive the data from the remote system.

You do not need to issue an invite function before a read operation to receive data. However, if an invite is outstanding for a program device to which a read is issued, the read completes the invite and receives the data.

Refer to Chapter 5 for additional information about the read and read-from-invited-program-

devices operations and their relationship to the invite function.

Timer (TIMER)

Your program can use the timer function before performing some specified function, such as a read-from-invited-program-devices operation. The timer function specifies an interval of time (in hours, minutes, and seconds) to wait before your program receives a timer-expired (0310) return code.

Use the TIMER keyword to set the timer for the specified interval of time. The TIMER keyword is issued on an output operation.

The format of the TIMER keyword is:

```
TIMER(HHMMSS | &field-name)
```

The parameter specified with the TIMER keyword can be one of the following:

HHMMSS

A literal value where HH is the number of hours, MM is minutes, and SS is seconds.

&field-name

A value where the field contains the TIMER value in the same HHMMSS format.

Your program continues to run, and all operations and functions are valid during the time interval. Your program must issue a read-from-invited-program-devices operation some time after it has issued the timer function, so it can accept the return code indicating that the timer interval has ended.

Only one time interval can be maintained for your program. If a previous timer function has been issued and the timer has not yet ended, the old time interval is replaced by the new interval.

The timer function can be used to vary the maximum amount of time that a read-from-invited-program-devices operation will wait for a response. When the time interval set by the TIMER keyword is in effect, the value specified for the WAITRCD parameter on the CRTICFF command is ignored.

You can use the timer function to retry other operations that may not be successful, possibly because of a temporary lack of resources (for example, during an acquire operation). To do this, issue the timer function, and then perform read-from-invited-program-devices operations until the timer interval ends. (The read-from-invited-program-devices operation allows the program to continue receiving input from other invited program devices while waiting for the timer.)

Refer to Chapter 5 for additional information on the read-from-invited-program-devices operation and its relationship to the timer function.

Record-Identification (RECID)

The RECID keyword identifies and selects the record format to use with an input operation based on the data received from the remote program. This keyword is applicable only if you specify FMTSLT(*RECID) on the ADDICFDEVE or OVRICFDEVE command.

The format of the RECID keyword is:

```
RECID(starting-position compare-value)
```

Specify the starting-position parameter as either nnnnn or *POSnnnnn, where nnnnn defines the beginning position of the compare value within the record format. The first position in the record is position 1.

Specify the compare-value parameter as:

**ZERO*. The data character in the position specified must be 0 (hex F0) to match the record identifier.

**BLANK*. The data character in the position specified must be a blank (hex 40) to match the record identifier.

'literal'. The data received, beginning with the position defined by the starting-position value, must match the literal specified here.

If the length of the record received is less than the number of positions examined for RECID value, the positions past the end of the record are treated as if they contained blanks. If the RECID keyword compare value specifies blanks for those positions, the data is considered a match.

For example, if your program receives both header and detail records from the remote program, you can specify the following in your ICF file:

```
RECID(1 'H')
RECID(1 'D')
```

Your program issues input operations to the file without specifying a record format name. You do not specify a record format name because the correct record format is not known until the data is received. Your program receives the records (either headers or detail) in the order they are sent by the remote program. For this example, the sending and receiving programs provide for an explicit code (an H for header records and a D for detail records) to identify the type of record being sent and received. The RECID keyword identifies the input buffer location where the H or D appears, and specifies the value (starting in the position specified) that identifies the record type.

The remote program must identify the type of record (either header or detail) by placing H or D in the first position of the data buffer.

For each input operation, the value specified in the first position of the buffer is compared with the value specified on the RECID keyword. If the value in a record is H, the format associated with the RECID(1 'H') specification is selected. Duplicate RECID keyword compare values are not checked. The first format with a compare value that matches the received value is used.

Be careful how you specify more than one RECID keyword within a file if more than one compare value begins in the same record position. For example, the following compare values begin in the same position:

```
RECID(1 'A')
RECID(1 'AB')
RECID(1 'ABC')
```

The first format is always selected if the data starts with an A, because any received records matching the last two compare values also match the first. Specify the longest value first to prevent confusion.

```
RECID(1 'ABC')
RECID(1 'AB')
RECID(1 'A')
```

You can use the RECID DDS keyword to eliminate processing of alphameric data in fields that should contain only numeric data. Refer to "Input Considerations" on page 8-3 for more information on eliminating data decimal errors.

Problem Notification

Use the fail, cancel, and negative-response functions to inform the application program of an error that has occurred in the data being sent or received. The DDS keywords associated with these functions are specified on an output operation.

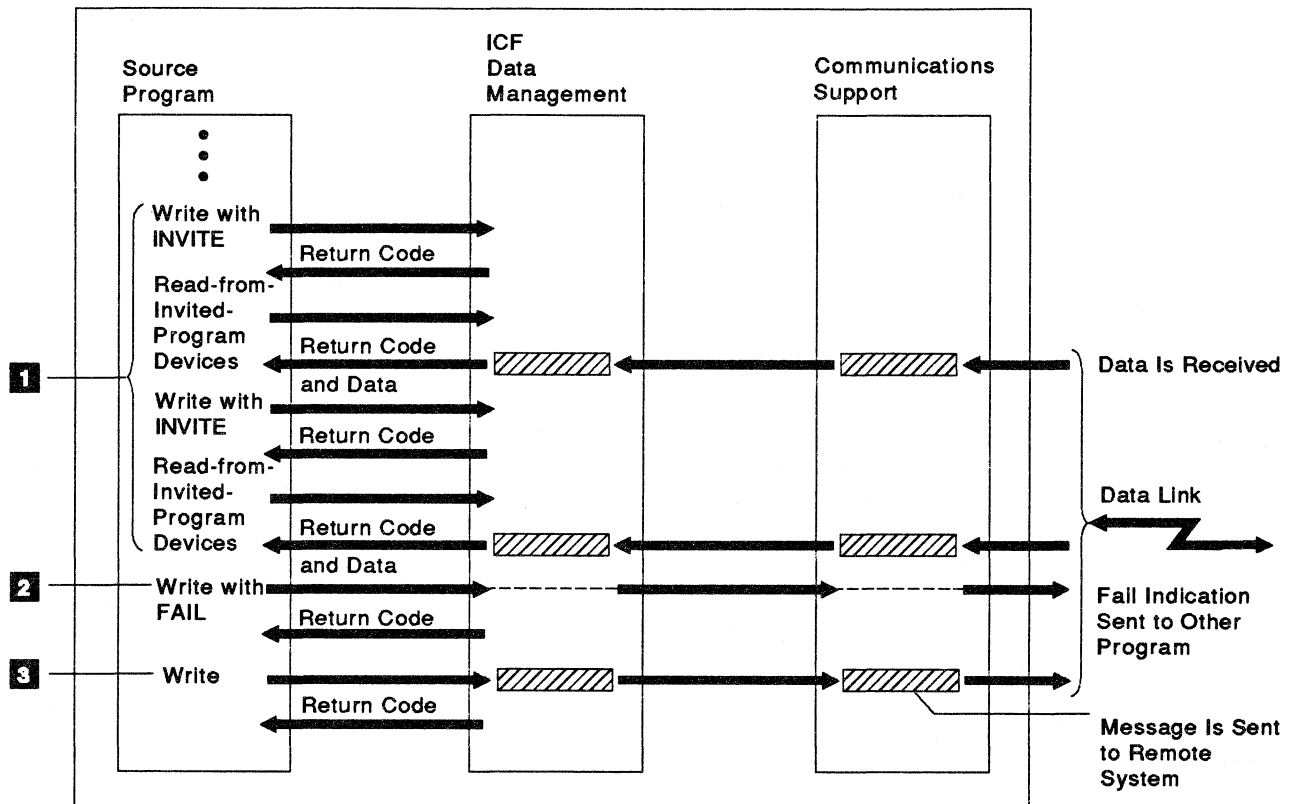
Fail (FAIL)

Use the FAIL keyword to indicate that an error has occurred when sending and receiving data.

If a program that is sending data issues a fail, it may indicate that the data just sent was in error. Your program can continue to send data and is usually responsible for the first error recovery. The communications type you are using determines whether data that is in the output buffer before issuing the fail function is sent to the remote system with the fail indication. The communications type determines the type of notification sent.

You can also use a fail function if your program receives data and detects an error in the received data. Figure 6-4 on page 6-10 shows how to use the fail function when your program is receiving data and detects an error.

AS/400 System



RSL5134-6

Figure 6-4. Using the FAIL Keyword to Send an Error Indication

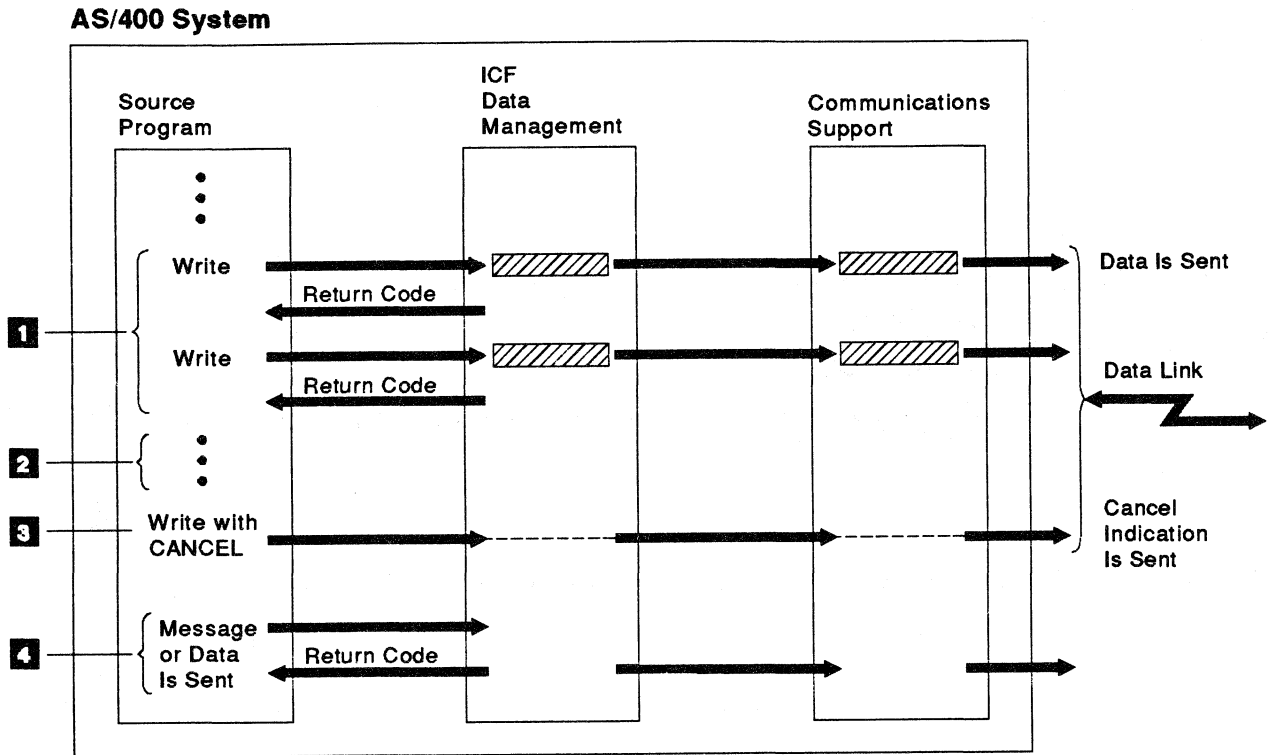
- 1 Your program is receiving data from the remote program.
- 2 While receiving data, your program determines that it must send a fail indication to the other program.
- 3 A message or data is then sent (write operation) to tell the other program why you sent the fail indication.

Refer to the RCVFAIL keyword described in "Using Response Indicator Keywords" on

page 6-18 for information on handling receipt of a fail indication.

Cancel (CANCEL)

Use the CANCEL keyword to tell the remote system to cancel the group of records you are currently sending. Your program can use the cancel function only when sending data (similar to issuing a fail when your program is sending data). Figure 6-5 on page 6-11 shows how to use the cancel function when a program is sending data and detects an error.



RSL5132-6

Figure 6-5. Using the CANCEL Keyword to Send an Error Indication

- 1 Your program is sending data to the remote system.
- 2 Your program checks the data and determines that something is wrong with it.
- 3 Your program uses a cancel function to tell the remote system to discard the data you have sent.
- 4 Your program can send a message indicating the problem, send the data again, receive more data, or end the transaction.

Refer to the RVCANCEL keyword described in "Using Response Indicator Keywords" on page 6-18 for information on handling receipt of a cancel indication.

Negative-Response (NEGRSP)

Use the NEGRSP keyword to tell the remote system that the data just received is not correct. The format of the NEGRSP keyword is:

```
NEGRSP[(&field-name)]
```

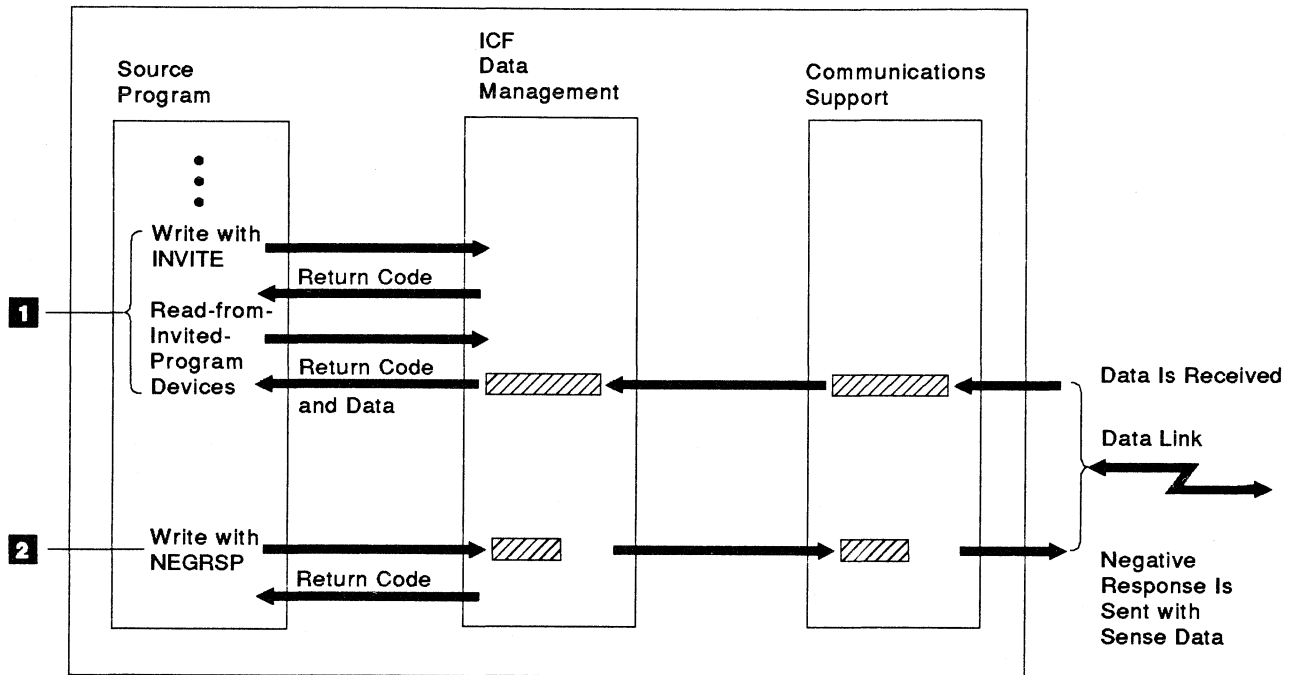
The optional parameter on the NEGRSP keyword specifies the name of the field that contains sense data to be sent to the remote program with the negative response.

Issuing a negative-response function is similar to issuing a fail function while receiving data, except that you can also include 8 characters of sense data with the negative-response function. The sense data tells the remote system what is wrong with the data you received. The first 4 characters of the sense data must begin with 10XX, 08XX, or 0000. The last 4 characters are user-defined. Refer to the appropriate communications programming manual for the communications type you are using for more information about the allowed sense values.

The sense data is sent in the normal output buffer. No other data is allowed to be sent with a negative-response function.

Figure 6-6 on page 6-12 shows how to send a negative response with a sense code to the remote system.

AS/400 System



RSL5130-5

Figure 6-6. Sending a Negative Response with Sense Code to Remote System

- 1** Your program finds that the data it is receiving is not correct.
- 2** The program sends a negative response to the remote system, including the sense data 08110000. The negative response tells the remote system that the data received is wrong, and the sense data 08110000 asks the remote system to cancel the current group of data records.

Refer to the RCVNEGRSP keyword described in "Using Response Indicator Keywords" on page 6-18 for information on handling receipt of a negative response indication.

Additional Keywords

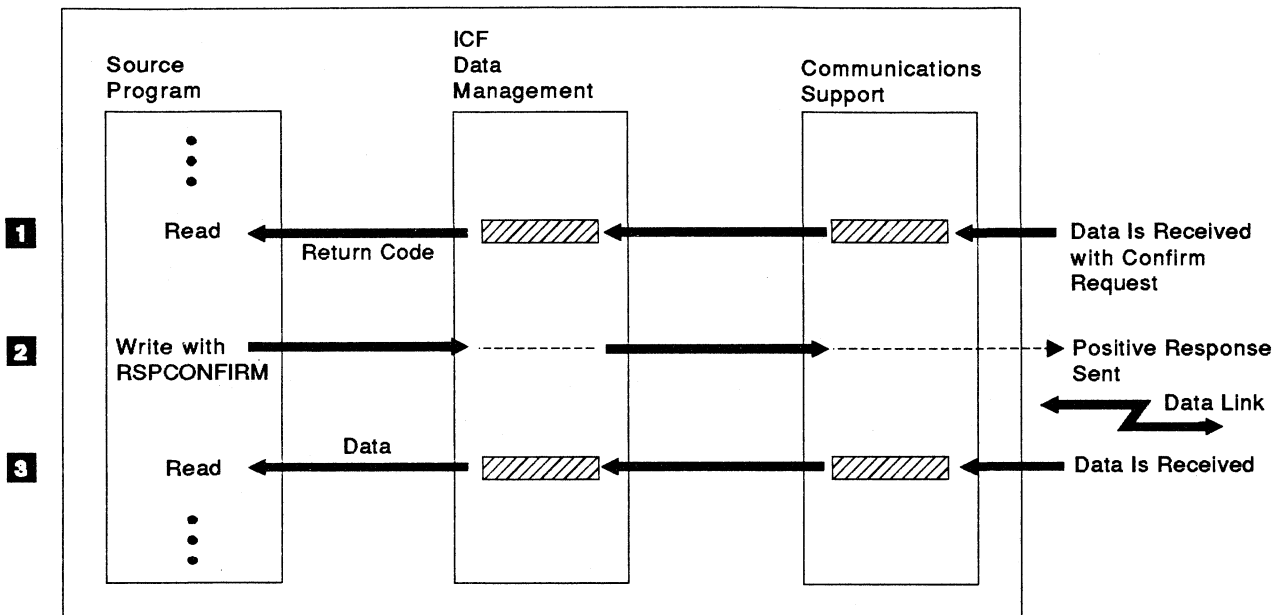
You can use the respond-to-confirm, request-to-write, allow-write, and cancel-invite functions to perform additional functions.

Respond-to-Confirm (RSPCONFIRM)

Use the RSPCONFIRM keyword to send a positive response to a received confirm request. The respond-to-confirm function can be used only when a confirm request is outstanding. You can check the major/minor return codes or use the RCVCONFIRM indicator to determine when to issue a respond-to-confirm function. After sending the response, your program can continue processing as indicated by any other information received.

Figure 6-7 on page 6-13 shows how to use the respond-to-confirm function.

AS/400 System



RSL5679-2

Figure 6-7. Using the Respond-to-Confirm Function

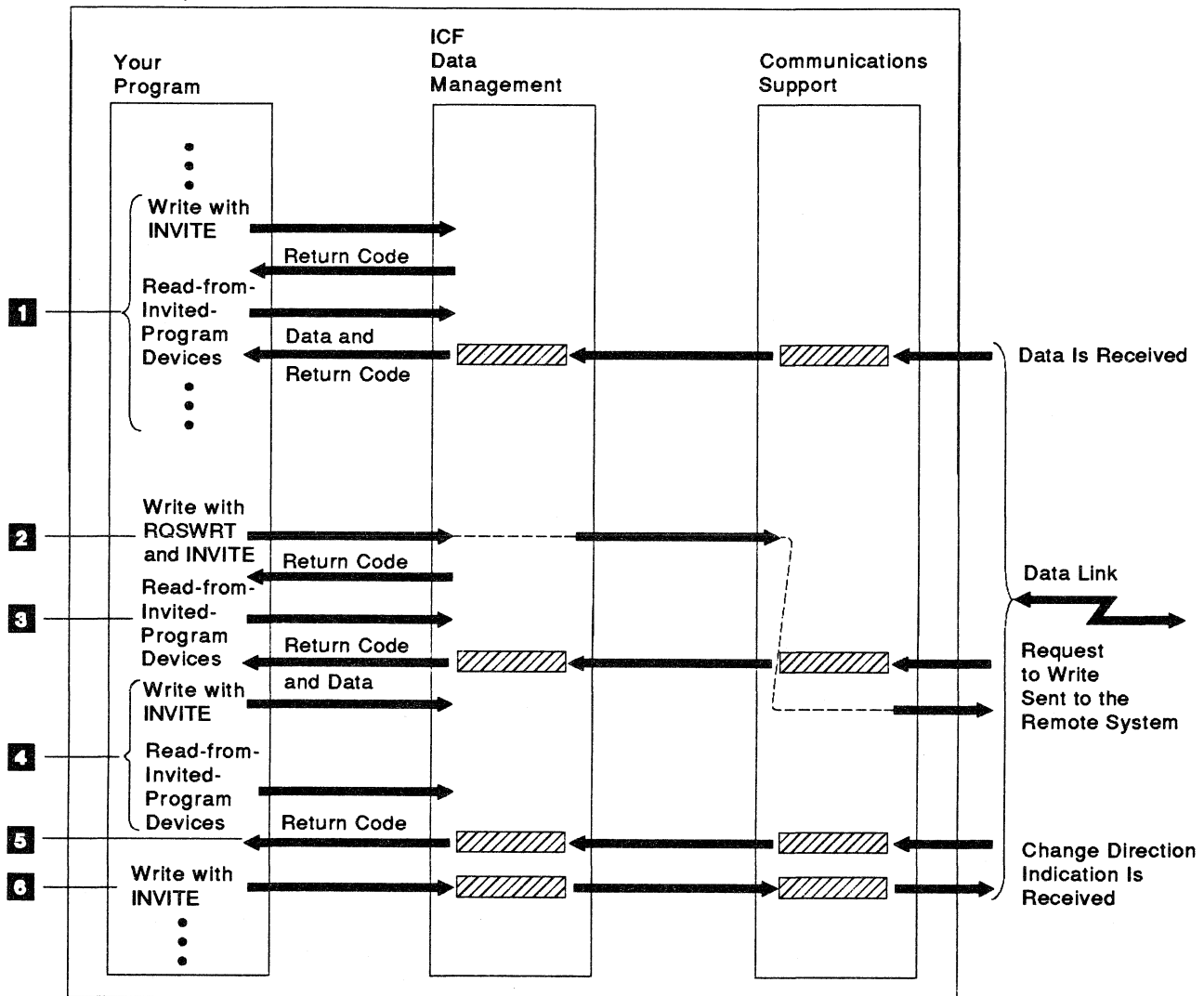
- 1** Your program is receiving data from the remote system. The return code indicates data and a confirm request. The read could have also been done with the RCVCONFIRM keyword to indicate that a confirm request was received.
- 2** Your program issues a write operation with the RSPCONFIRM keyword in effect to acknowledge the receipt of data.
- 3** Your program continues to receive because the remote program is still in send state.

Request-to-Write (RQSWRT)

Use the RQSWRT keyword, while your program is receiving data, to ask the remote system to stop sending so your program can send. The request-to-write function tells the remote system you want to change the direction of data transmission. If the remote system allows the change, your program can send either data or a message, or both, to the remote system. After issuing the request-to-write, your program must continue receiving data until the remote system sends a notification indicating it is ready to receive.

Figure 6-8 on page 6-14 shows how to use the request-to-write function.

AS/400 System



RSL5128-5

Figure 6-8. Using the Request-to-Write Function

- 1 Your program is receiving data from the remote system. The program processes the data received, then receives data again.
- 2 At some time while data is being received, your program determines that it needs to send a message to the remote system. Your program issues a write operation, with the RQSWRT and INVITE keywords in effect, to ask the remote system to stop sending so your program can send the message. The request-write indication is sent to the remote system at the first available opportunity. Since the session is in receive state, the indication may be held until the next data record is received.
- 3 After issuing the request-to-write function, your program must continue receiving data until it gets a return code indicating that the remote system is ready to receive. To continue receiving, another read-from-invited-program-devices operation is used.
- 4 Another invite and read-from-invited-program-devices operation is issued to continue receiving data.
- 5 When the remote system is ready to receive, it sends one more data record with a change-direction indication. The record says the remote system is now ready to receive data or, as in this example, a message.

- 6** A write operation with the INVITE keyword in effect is used to send the message to the remote system and ask the remote system to continue sending data.

When your program receives a request-to-write request from the remote system, a code is set in the I/O feedback file-dependent section. Refer to Figure C-5 on page C-5 for more information about where this field is in the I/O feedback area.

The code indicates the following conditions:

- 0** Continue sending as normal.
- 1** A request-to-write has been received.

Refer to the appropriate communications programmer's manual for the communications type you are using for more specific information on what this code means for the communications type you are using.

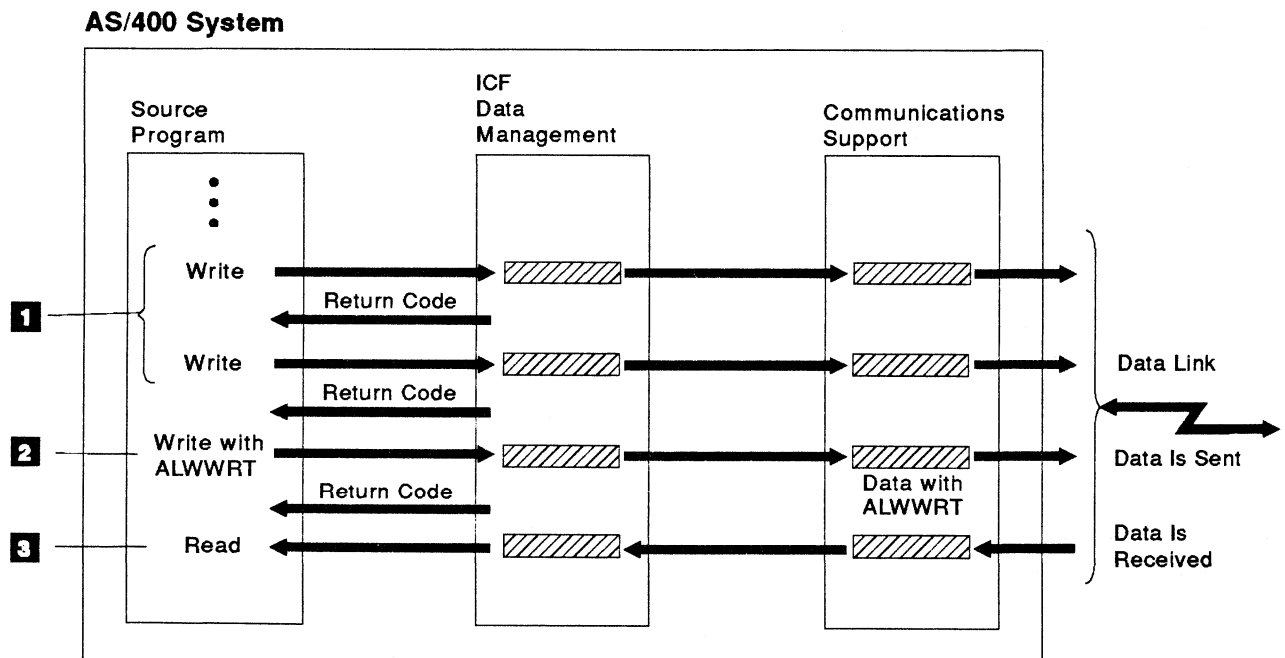
Allow-Write (ALWWRT)

Use the ALWWRT keyword to explicitly inform the remote system that your program is done sending. The allow-write function clears the buffers, forcing any data to be sent. The same function occurs automatically if you issue an input operation after a write operation. In that case, the ALWWRT DDS keyword is not required. After issuing an allow-write, your application program can issue an input operation to receive data from the remote system.

Figure 6-9 shows how to use the allow-write function.

- 1** Your program sends several data records to the remote system.
- 2** You use an allow-write with the last record to inform the remote system you are done sending.
- 3** The remote system can now send data, and your program must begin receiving.

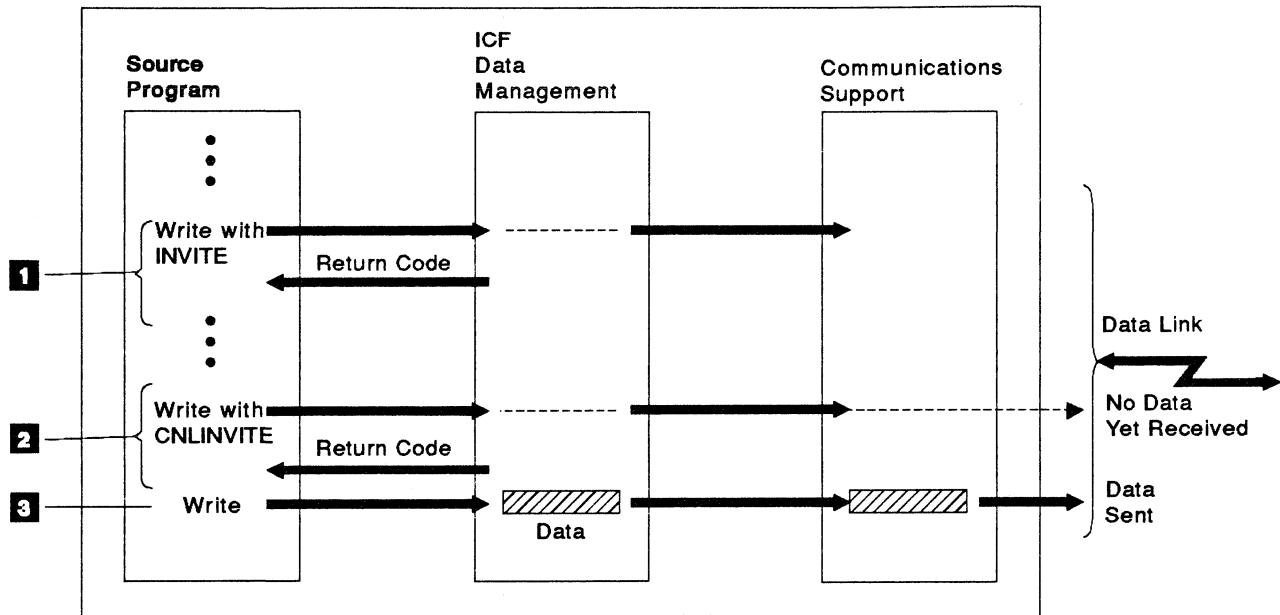
Refer to "Receive-Turnaround" on page 6-19 for information on handling receipt of an allow-write indication.



PLS182-3

Figure 6-9. Using the Allow-Write Function

AS/400 System



RSL5183-5

Figure 6-10. Using the Cancel-Invite Function

Cancel-Invite (CNLINVITE)

Use the CNLINVITE keyword to cancel a valid invite for which no data has yet been received from an invited program device. Your program can continue to send data.

Figure 6-10 shows how to use the CNLINVITE keyword.

- 1** Your program issues an invite operation to receive data from the remote program, then continues processing.
- 2** Your program uses the cancel-invite function to cancel the previous invite operation. Your program must check the return code it receives to determine if any data has already been received from the remote system.
- 3** Your program can continue to send data if data was not received.

Ending a Communications Transaction

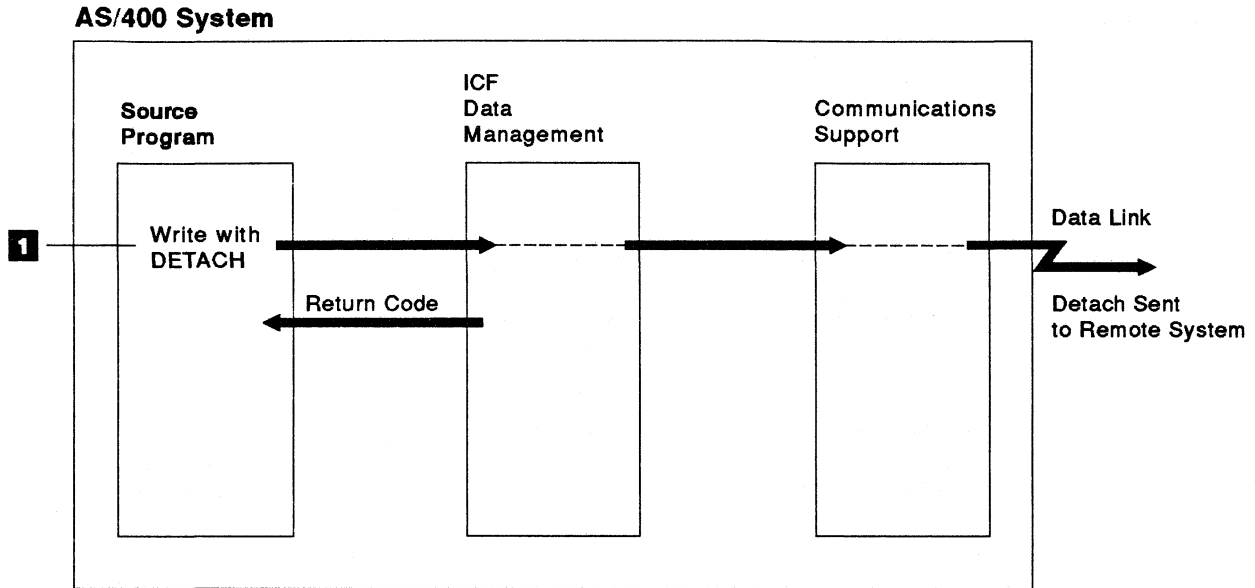
A communications transaction can be ended by your program or by the program at the remote system. Your job and the remote system that your system is communicating with determine the program that ends the transaction.

Communications with the remote program end when your program ends the transaction. However, the session may still exist if your program started the session. If the session still exists, you can end the session or you may be able to start another program at the remote system.

Detach (DETACH)

Use the DETACH DDS keyword to end the transaction. The detach explicitly informs the remote program that your program is done sending and has ended the transaction.

Figure 6-11 on page 6-17 shows how your program can end a communications transaction.



RSL5136-5

Figure 6-11. Ending a Communications Transaction

- 1 Your program issues the detach to tell the remote system that your program ended the communications transaction.

Refer to the RCVDETACH keyword described in "Using Response Indicator Keywords" on page 6-18 for information on handling receipt of a detach indication.

If a detach function is issued by the target program, the session is ended as well as the transaction.

Ending the Communications Session

How the communications session is ended depends on whether your program or the remote system started the session.

The release operation ends the session only if

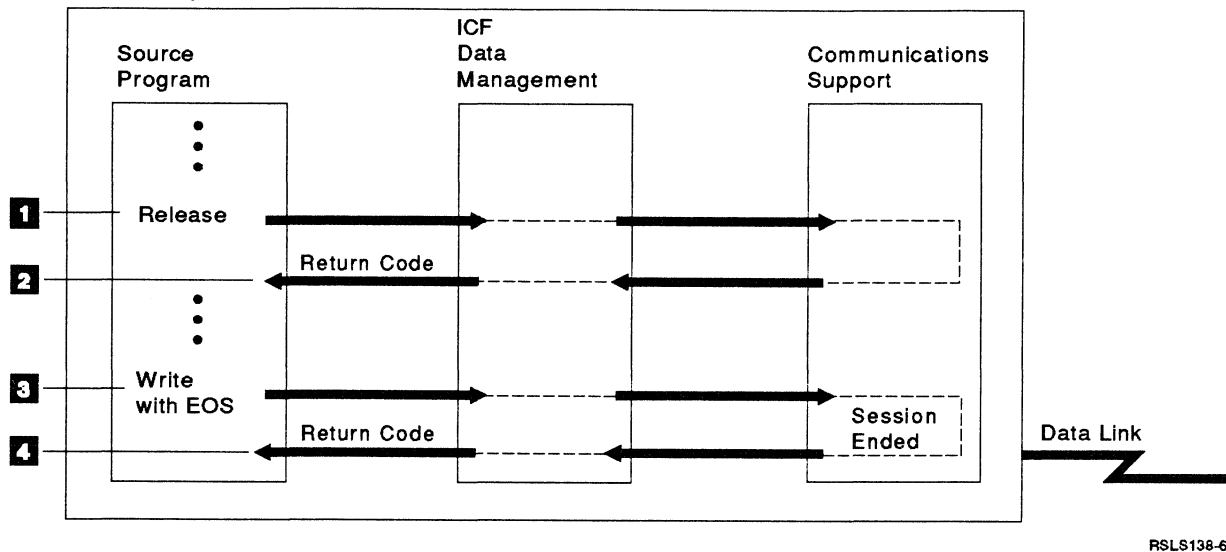
all processing is complete. The end-of-session operation *always* ends the session.

End-of-Session (EOS)

Use the EOS DDS keyword to issue an end-of-session function. The only possible return codes from end-of-session are 0000 or 830B (program device not acquired).

If the target program ends the transaction by the detach function, the session is ended implicitly. If the source program ends the transaction, the target program must issue an end-of-session or go to the end of the job to end the session. Figure 6-12 on page 6-18 shows how you can end the session using the release operation and the end-of-session function.

AS/400 System



RSL5138-6

Figure 6-12. Using the Release and End-of-Session Functions

- 1 Your program issues the release operation to end the current communications session.
- 2 The return code tells your program whether the session was ended, or if an error occurred while trying to end the session. If, for example, all transactions have not ended when the release operation is issued, an error occurs and the session is not ended.
- 3 If an error occurs and normal recovery is not possible, your program can use the end-of-session function to end the session.
- 4 The end-of-session function always ends the session.

They have no effect on an output operation. Multiple response keywords can be used on a single input operation.

Receive-Confirm

Use the RCVCONFIRM keyword to request that a response indicator be set on if the record received from the remote system contains a confirm request. A received confirm request indicates that the remote program expects your program to do a specific action to synchronize the programs. The action can be a write with the RSPCONFIRM keyword (positive response), or a write with the FAIL keyword (negative response). If the session is abnormally ended (end-of-session or job end before ending the transaction), a negative response is sent.

This same type of information can be determined by checking the major/minor return code returned in the I/O feedback area at the completion of each operation.

The program receiving the confirm indication is responsible for making sure that the response (positive or negative) is returned to the program requesting the confirmation.

If you want to return a positive response to the remote system, issue a write with the RSPCONFIRM DDS keyword in effect. If you want a negative response returned, either issue a write with the FAIL DDS keyword in effect, or

Using Response Indicator Keywords

Response keywords provide information to your program about the data record being received or the actions taken by the remote program. Check which response indicators are set when your program does an input operation to determine:

- What the remote program sent
- What the remote program expects from your program
- What your program's next operation should be

Response keywords are only effective for input operations or a combined output then input.

abnormally end the session (end-of-session or job end before ending the transaction).

Receive-End-of-Group

Use the RCVENDGRP keyword to request that a response indicator be set on if the record received from the remote system contains an end-of-group indicator. The response indicator is set if the last record received in the input buffer was the end of a user-defined group of records.

Receive-Function-Management-Header

Use the RCVFMH keyword to request that a response indicator be set on if the record received from the remote system contains a function-management-header indication. The response indicator is set if the data received in the input buffer is function-management-header data. Asynchronous, finance, intrasystem, and retail communications give the user data along with the function-management-header indication in one operation. If you are using SNUF, however, you must do an additional input operation to get the remaining user data that accompanied the function-management-header.

Receive-Fail

Use the RCVFAIL keyword to request that a response indicator be set on if the record received from the remote system contains a fail indication. The remote program informs your program that it found something wrong while sending or receiving data. Your program should issue an input operation after receiving a fail indication. The program sending the fail indication must start the error recovery.

Receive-Cancel

Use the RCVCANCEL keyword to request that a response indicator be set on if the record received from the remote system contains a cancel indication. The remote system informs your program that the current chain of data is

not correct. Your program should discard the data, then continue to receive or end the job.

Receive-Negative-Response

Use the RCVNEGRSP keyword to request that a response indicator be set on if the data received from the remote system contains a negative-response indication. The remote system informs your program that an error was detected in the data it just received. You may receive an 8-byte sense code with the negative response signal.

Receive-Turnaround

Use the RCVTRNRND keyword to request that a response indicator be set on if the data received from the remote system contains a change-in-transmission-direction indication. The remote system informs your program that it is finished sending data, and is ready to receive data. Your program can begin sending data.

Receive-Detach

Use the RCVDETACH keyword to request that a response indicator be set on if the record received from the remote system contains a detach indication. The remote system informs your program that it is ending this communications transaction with your program. Your program can no longer communicate with the remote program. The session with the remote system may still exist if your program started the session. If the remote system started the session, communications with the remote system are ended.

Example DDS Files for Creating an Intersystem Communications Function File

Figure 6-13 on page 6-20 and Figure 6-14 on page 6-21 are DDS source files that can be used to create an ICF file. Files created using this source DDS are used in the application program examples in Chapter 9 through Chapter 11.

```

A*****
A*
A*          ICF FILE          *
A*          USED IN BATCH DATA TRANSFER PROGRAM      *
A*
A*****
A*
A* FILE LEVEL INDICATORS:
A*
A          INDARA
A*
A          RCVTRNRND(15 'END OF DATA')
A*
A 30      DETACH
A*
A          INDTXT(30 '30->DETACH TAR-
A          GET PROGRAM. ')
A*
A          RCVDETACH(35 'RECEIVED -
A          DETACH. ')
A*
A*****
A*          ICF RECORD FORMATS          *
A*****
          R RCVDATA
          RCVFLD          80A
          R SNDDATA
          SNDFLD          80A
          R EVOKPGM
A 50      EVOKE(&LIB/&PGMID)
A          PGMID          10A P
A          LIB          10A P
A          R ENDREC
A          R INVITE
A 45      INVITE

```

Figure 6-13. DDS Source File for a Batch Data Transfer Program


```

300 A*****
400 A*
500 A*          ICF FILE
600 A*          USED IN SOURCE MULTIPLE SESSION PROGRAM
700 A*
800 A*****
900 A          INDARA
1000 A      R ITMRSP
1100 A          RECID(1 'I')
1200 A          RECITM          1
1300 A          ITEMNO          6 0
1400 A          DESC            30
1500 A          QTYLST          7 0
1600 A          QTYOH           7 0
1700 A          QTYO0           7 0
1800 A          QTYBO           7 0
1900 A          UNITQ           2
2000 A          PR01            7 2
2100 A          PR05            7 0
2200 A          UFRT            5 2
2300 A          SLSTM           9 2
2400 A          SLSTY           11 2
2500 A          CSTTM           9 2
2600 A          CSTTY           11 2
2700 A          PRO             5 2
2800 A          LOS             9 2
2900 A          FILL1           56
3000 A      R DTLRSP
3100 A          RECID(1 'C')
3200 A          RCVTRNRND(90)
3300 A          RECCUS          1
3400 A          CUSTNO          6 0
3500 A          DNAME           30
3600 A          DLSTOR          6 0
3700 A          DSLSTM          9 0
3800 A          DSPM01          9 0
3900 A          DSPM02          9 0
4000 A          DSPM03          9 0
4100 A          DSTTYD          11 0
4200 A          IDEPT           3 0
4300 A          FILL2           57
4400 A      R DETACH
4500 A          DETACH
4600 A      R EOS
4700 A          EOS
4800 A      R EVKREQ
4900 A          EVOKE(&LIB&PGMID)
5000 A          PGMID           10A P
5100 A          LIB             10A P
5200 A      R ITMREQ
5300 A          INVITE
5400 A          ITEMNO          6 0
5500 A      R DTLREQ
5600 A          INVITE
5700 A          CUSTNO          6 0
          *****
          * * * * END OF SOURCE * * * *

```

Figure 6-14. DDS Source File for a Multiple Session Program

The following is an example of a Create Inter-system Communications Function (CRTICFF) command used to create an ICF file from a DDS source file:

```

CRTICFF FILE(ICFLIB/ICFFILE) SRCFILE(ICFLIB/QDSSSRC)
SRCMBR(*FILE) ACQPGMDEV(*NONE) MAXPGMDEV(10)
TEXT('ICF FILE EXAMPLE')

```

The created file has the following attributes:

- The file name is ICFFILE and it is stored in library ICFLIB, as specified on the FILE parameter.

- The SRCFILE parameter indicates that the DDS source file, from which this ICF file is created, is a member in file QDSSRC in library ICFLIB.
- The SRCMBR parameter indicates the source file has the same name as the file you are creating.
- The ACQPGMDEV parameter indicates that no program device is automatically acquired

when the file is opened. Your program must explicitly issue the acquire.

- The MAXPGMDEV parameter indicates that up to 10 program devices can be acquired and active with this file.
- The TEXT parameter describes the file.

The remaining parameters not specified on the CRTICFF command are assigned default values. Refer to Chapter 4 and the *CL Reference* manual for more information on these parameters and their default values.

The following is an example of an ADDICFDEVE command used to add a program device entry to the ICF file just created:

```
ADDICFDEVE FILE(ICFLIB/ICFFILE) PGMDEV(PGMDEVA)
RMTLOCNAME(CHICAGO) FMTSLT(*PGM)
```

The file now has a program device entry with the following attributes:

- The PGMDEV parameter indicates that PGMDEVA is the program device name added to the file. This is the program device name used in your program.
- The RMTLOCNAME parameter indicates that CHICAGO is the name of the remote location

associated with PGMDEVA. CHICAGO is the remote location name specified on the device description when you configured your system for communications.

- The FMTSLT parameter indicates that *PGM is the format selection option used on input operations. For more information on this parameter, refer to “Format Selection Processing” on page 5-9.

The remaining parameters not specified on the ADDICFDEVE command have assigned default values. Refer to Chapter 4 and the *CL Reference* manual for more information on these parameters and their default values.

Keyword Processing Charts

Figure 6-15 and Figure 6-16 on page 6-23 summarize the DDS keywords discussed in this chapter. Use these charts for a quick reference when defining and creating an ICF file, and when writing application programs.

Figure 6-15 lists the DDS keywords defined in this chapter that are supported by the various communications types for output operations.

Figure 6-15 (Page 1 of 2). Output DDS Processing Keyword Support

DDS Keyword	APPC	SNUF	BSCCL	Asynchronous	Intrasystem	Finance	Retail
ALWWRT	X	X	X		X		
CANCEL		X			X	X ²	X
CNLINVITE		X	X	X	X	X	X
CONFIRM	X				X		
DETACH	X	X	X	X ¹	X		X
ENDGRP		X	X		X	X	X
EOS	X	X	X	X	X	X	X
EVOKE	X	X	X	X	X		X
FAIL	X	X	X	X	X	X	
FMH		X		X ¹	X	X	X
FMTNAME	X				X		
FRCDTA	X				X	X	X
INVITE	X	X	X	X	X	X	X
NEGRSP		X			X	X	X
RQSWRT	X	X	X		X		
RSPCONFIRM	X	X			X		
SECURITY	X	X	X	X	X		

Figure 6-15 (Page 2 of 2). Output DDS Processing Keyword Support

DDS Keyword	APPC	SNUF	BSCSEL	Asynchronous	Intrasystem	Finance	Retail
SUBDEV			X		X		
SYNLVL	X				X		
TIMER	X	X	X	X	X	X	X
VARBUFMG ³	X						
VARLEN	X	X	X	X	X	X	X

- 1 Use of these keywords are restricted. Refer to the *Asynchronous Communications Programmer's Guide* for more details.
- 2 This keyword is not valid for the 3694 controller. Refer to the *Finance Communications Programmer's Guide* for more details.
- 3 Use of this keyword is restricted. Refer to the *APPC Programmer's Guide*.

Figure 6-16 lists the DDS keywords defined in this chapter that are supported by the various communications types for input operations.

Figure 6-16. Input DDS Processing Keyword Support

DDS Keyword	APPC	SNUF	BSCSEL	Asynchronous	Intrasystem	Finance	Retail
RCVCANCEL		X			X		X
RCVCONFIRM	X	X			X		
RCVDETACH	X	X	X		X		X
RCVENDGRP		X	X		X	X	X
RCVFAIL	X			X	X		
RCVFMH		X			X	X	X
RCVNEGRSP		X			X	X	X
RCVTRNRND	X	X	X		X		
RECID	X	X	X	X	X	X	X

Figure 6-17 shows the priority sequence used by ICF in processing these DDS keywords and data during output operations.

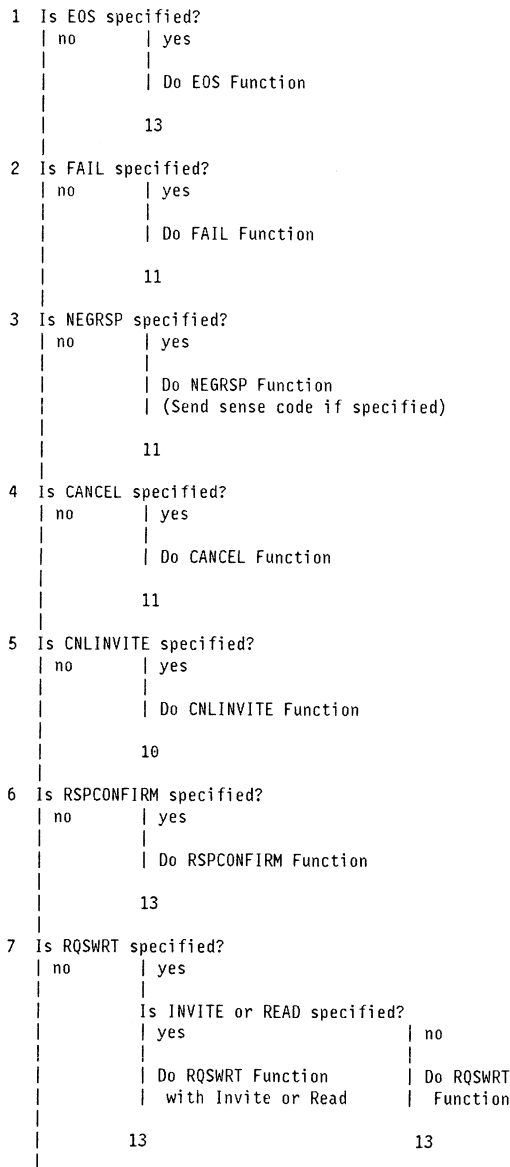
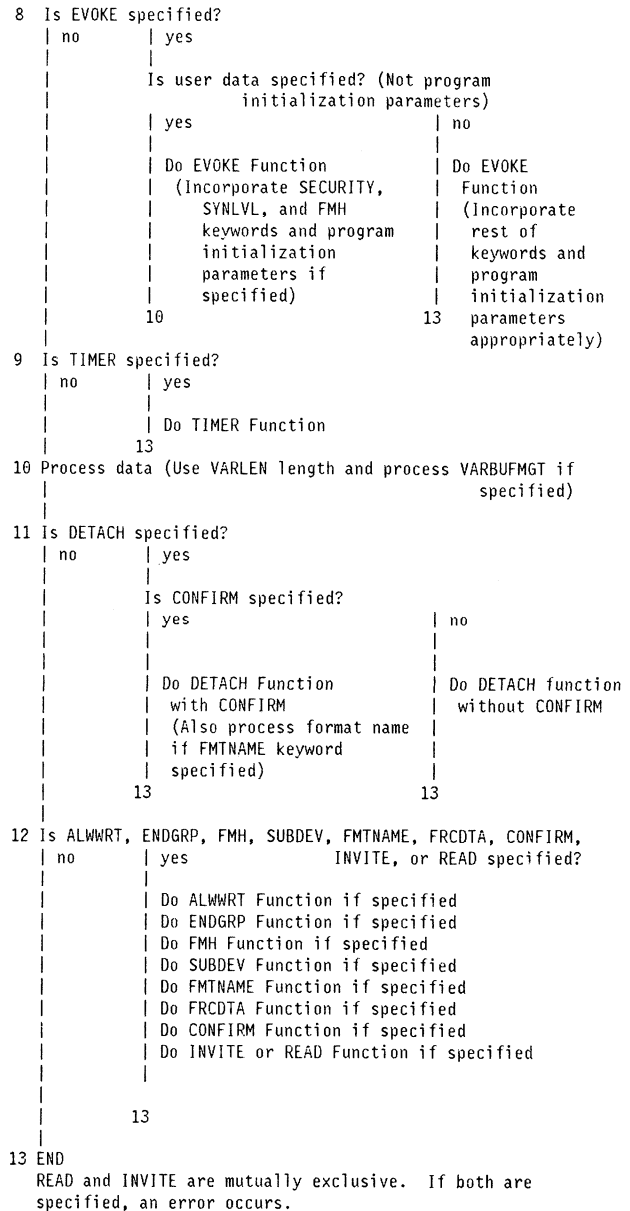


Figure 6-17 (Part 1 of 2). Keyword Processing Chart



The rest of the keywords: VARBUFMTG, VARLEN, DETACH, CONFIRM, INVITE, ALWVRT, ENDGRP, FMH, SUBDEV, FMTNAME, and FRCDTA, are processed in the same priority sequence as step 11 and step 12 above. SUBDEV is not allowed with EVOKE if there is no user data.

Figure 6-17 (Part 2 of 2). Keyword Processing Chart

Chapter 7. Using System-Supplied Communications Formats

This chapter defines the system-supplied communications formats you can use in your program to control data communications with the remote system. These system-supplied formats are used in place of user-defined data description specifications (DDS) record formats on the write operation. This chapter also maps these system-supplied communications formats to their DDS keyword counterparts in Figure 7-24 on page 7-23.

Programming examples are included to show you how these system-supplied formats are used. These examples are program segments only. You can find complete C/400, COBOL/400, and RPG/400 programming examples in Chapter 9 through Chapter 11.

All system-supplied formats described in this chapter may not be supported by the communications type you are using. Figure 7-23 on page 7-22 summarizes the support provided by each communications type. For more detail, refer to the appropriate communications programming manual for the communications type you are using.

General Description

You can use system-supplied formats for communications only when using an ICF file. You can either create your own file or use the default file provided by ICF for communications when using system-supplied formats. This file, QICDMF, is in library QSYS. You must still perform the override commands for QICDMF to define your program device names.

The QICDMF file was created with the following characteristics:

- The INDARA keyword is used in this file; therefore, a separate indicator area must be specified in your program when this file is used.
- *NONE was specified for the ACQPGMDEV parameter. Therefore, no program device is acquired when the file is opened.
- The maximum record length for the file is 4096 bytes. The maximum record length is

used in allocating I/O buffers. If your program does not need this large a record, you may want to override this value by using the Override Intersystem Communications Function File (OVRICFF) command, specifying the MAXRCLEN parameter.

- The maximum number of program devices that can be acquired with this file is five. If your program uses more than five program devices, you will need to change this file by using the Change Intersystem Communications Function File (CHGICFF) command and specifying a larger value for the MAXPGMDEV parameter.
- 30 SECONDS was specified for the WAITFILE parameter.
- *NOMAX was specified for the WAITRCD parameter.
- *NO was specified for the SHARE parameter.
- *USE was specified for the AUT parameter. Refer to the *Security Concepts and Planning* for information on what rights this characteristic provides.

Do not change this file with the CHGICFF command unless you need to change the maximum number of program devices or want to provide different default characteristics system-wide than those provided at file creation. Use the Override Intersystem Communications Function File (OVRICFF) command to temporarily override any characteristics needed by a particular application.

Do not add any program device entries to the file using the Add Intersystem Communications Function Device Entry (ADDICFDEVE) command. Define program device entries using the Override Intersystem Communications Function Device Entry (OVRICFDEVE) command.

The primary communications functions you can perform using system-supplied formats are:

- Evoke functions (starting remote programs)
- Output functions (sending data)
- Detach functions (ending communications transactions)
- End-of-session functions (ending the session)

These functions are described on the following pages.

All of the system-supplied formats are specified on output operations. The system-supplied formats that allow you to perform the invite and timer functions do, however, affect input processing.

Note: This chapter discusses only how you can use system-supplied formats to do specific communications functions, such as starting and stopping a communications transaction and sending data. Your program will, of course, need to perform additional operations such as starting a session and receiving data. Refer to the appropriate sections in Chapter 5 for information on these operations.

Starting a Program on the Remote System

The target program must be started before communications can begin between your program and a target program. To start a target program and to start a communications transaction, your program must issue an evoke function.

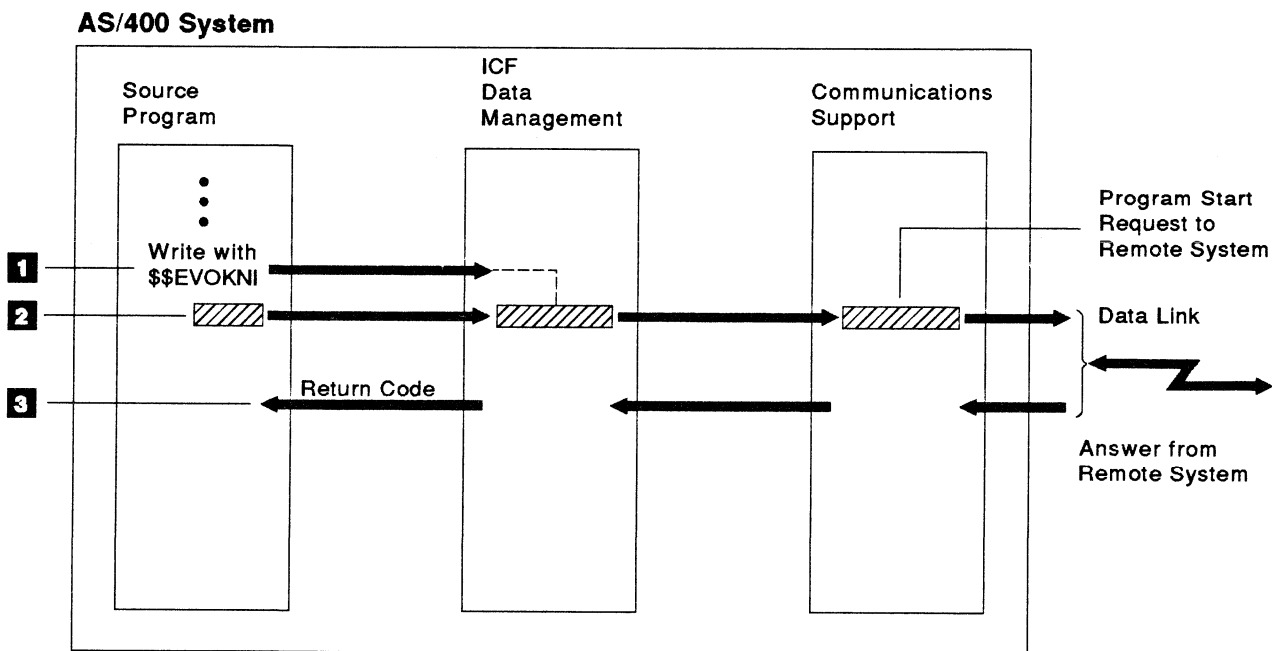
Evoke

You can use one of the following three system-supplied formats to perform an evoke function:

- **Evoke (\$\$EVOKNI).** Starts the specified program on the remote system. Your program remains in send state, so it can send data to the target program.
- **Evoke with Invite (\$\$EVOK).** Starts the specified program on the remote system and invites that program to send data.
- **Evoke with Detach (\$\$EVOKET).** Starts the specified program on the remote system and ends the communications transaction, without allowing the target program to communicate in return. Refer to “Ending a Communications Transaction” on page 7-18 for more information on the detach function.

Figure 7-1 shows how to start a target program on the remote system.

- 1 The source program uses an evoke function to start the program at the remote system.
- 2 The evoke parameters, including program name, library name, and security information, are sent to the remote system.



RSL670-2

Figure 7-1. Starting a Target Program

- 3 A successful completion return code tells the source program that the evoke function was accepted and a program start request was sent to the remote system. If the program start request is successful, both the program at the remote system and the communications transaction are started.

You must specify an **evoke parameter list** in the output buffer with an evoke function. The evoke parameter list contains information for the remote system, such as what program to start on the remote system. Specify the field parameters in that list using the format shown in Figure 7-2.

Figure 7-2. Evoke Parameter List

Positions	Field Description
1 through 8	The name of the program to be evoked (left-adjusted)
9 through 16	The password you use to sign on the remote system (left-adjusted)
17 through 24	The user identifier you use to sign on the remote system (left-adjusted)
25 through 32	The name of the remote system library that contains the program to be evoked (left-adjusted)
33 through 52	Reserved
53 through 56	The length of data (program parameters)
57 through xxxx	Program initialization parameters

If a field is not used, enter the correct number of blanks for the unused field.

If multiple program initialization parameters are used, the program is responsible for using the proper separation characters for the remote system. For example, if the remote system is an

AS/400 system, multiple parameters must be separated by a comma.

If the remote system is another AS/400 system, the program parameters are passed to the target program as if they were passed from a Call a Program (CALL) command. Data sent with an evoke function are parameters used by the target program.

System-supplied formats do not allow a synchronization level of CONFIRM and always revert to the default synchronization level of NONE.

The following is an example of a C/400 write statement that can be used to issue an evoke.

```

struct {
    char program_name??(8??);
    char password??(8??);
    char user_id??(8??);
    char library_name??(8??);
    char filler??(20??);
    char data_length??(4??);
    char data??(1000??);
} evoke_rec;

:
_RFILE *icffptr;          /* Pointer to the ICF file */
:
icffptr = _Ropen("ICFFILE","ab+ indicators=y riofb=y");
:
_Rformat(icffptr, "$$EVOKNI");
/* Set evoke w/no invite format */
_Rpgmdev(icffptr, "CM1");
/* Set default device to CM1 */
_Rwrite(icffptr, &evoke_rec, sizeof(evoke_rec));/* Do the evoke */

```

The following is an example of a COBOL/400 WRITE statement which can be used to issue an evoke.

```

01 DATA-RECORD.
03 PROGRAM-NAME    PIC X(8).
03 PASSWORD        PIC X(8).
03 USER-ID         PIC X(8).
03 LIBRARY-NAME    PIC X(8).
03 FILLER          PIC X(20).
03 DATA-LENGTH    PIC 9(4).
03 THE-DATA        PIC X(256).
.
.
WRITE TRANSACTION-RECORD FROM DATA-RECORD,
FORMAT IS '$$EVOKNI', TERMINAL IS ICF-PGMDEV.

```

Figure 7-3 on page 7-4 is an example of an RPG/400 output specification used to issue an evoke function.

Program	Keying Instruction	Graphic Key	Card Electro Number
Programmer	Date		

Page 1 of 2
Program Identification: 75 76 77 78 79 80

Line	Form Type	Filename or Record Name	Type (N/D/E)	Space	Skip	Output Indicators	Field Name or EXCPT Name	Edit Codes	End Position in Output Record	P/B/L/R	Constant or Edit Word																									
											Commas		Zero Balances to Print		No Sign		CR		-		x = Remove Plus Sign		Y = Date Field Edit		Z = Zero Suppress		5 - 9 = User Defined									
											Yes	No	Yes	No	1	2	A	B	J	K	L	M														
01	O	ICFFILE	E				EVOKE				1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24		
02	O							K18																												
03	O																																			
04	O																																			
05	O																																			
06	O																																			
07	O																																			
08	O																																			
09	O																																			

RSL5184-1

Figure 7-3. Evoke RPG/400 Output Specification

Sending Data

A data record can be sent from your program to the remote program. The following list describes each of the system-supplied send formats that can be used to send data:

- **Send (\$\$SENDNI)**. Sends one data record to the remote program.
- **Send with Invite (\$\$SEND)**. Sends one data record to the remote program and issues an invite to the remote program. Your program must use an input operation to receive the data sent from the remote system.
- **Send with Function-Management-Header (\$\$SENDNF)**. Sends a data record that includes a function-management-header to the remote program. Function-management-header data contains control information that tells the remote system about the data being sent.
- **Send with Function-Management-Header and Invite (\$\$SENDFM)**. Sends a data record that

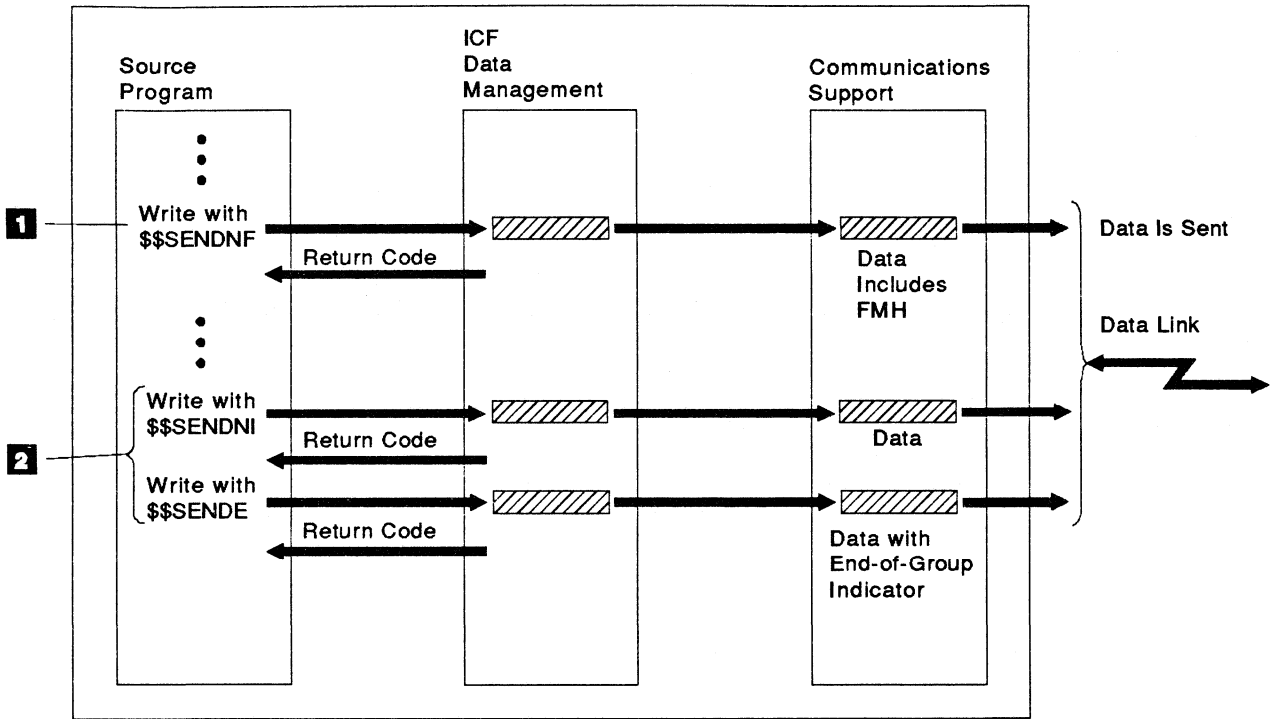
includes a function-management-header and an invite to the remote program.

- **Send with End-of-Group (\$\$SENDE)**. Sends a data record to the remote program and tells the remote program that the record is the last in a group or chain of records.
- **Send with Detach (\$\$SENDET)**. Sends a data record to the remote program and tells the remote program that your program is ending this communications transaction. Communications between the two programs have ended. Refer to “Ending a Communications Transaction” on page 7-18 for more information on the detach function.

Note: Except for \$\$SENDNF and \$\$SENDFM, you can specify a length of zero and perform any of the preceding functions without sending any data.

Figure 7-4 on page 7-5 shows how to use system-supplied formats to send data.

AS/400 System



RSL5671-3

Figure 7-4. Using \$\$\$SENDNF, \$\$\$SENDNI, and \$\$\$SENDE to Send Data

- 1 Your program sends a record to the remote system and, with the \$\$\$SENDNF communications format, indicates that the first part of the data is function-management-header data. The function-management-header data contains information about the user data that follows.
- 2 Your program continues sending data records to the remote system. Your program uses the \$\$\$SENDE communications format on the last record to indicate it is the last in this group of records.

```

struct {
  char record_length??(4??);
  char data??(80??);
} data_rec;

:
:
| _RFILE *icffptr;          /* Pointer to the ICF file */
:
:
| icffptr = _Ropen("ICFFILE","ab+ indicators=y riofb=y");
:
:
| _Rformat(icffptr, "$$SENDNF");
| /* Set write w/no invite format */
| _Rpgmdev(icffptr, "CM1"); /* Set default device to CM1 */
:
:
| strncpy(data_rec.record_length, "0080", 4); /*Set record length*/
| _Rwrite(icffptr, &data_rec, sizeof(data_rec)); /* Do the write */

```

Each of the preceding functions requires the fields shown in Figure 7-5 in the output buffer.

Figure 7-5. Required Output Fields

Positions	Description
1 through 4	Length of user data (in decimal)
5 through xxxx	The user data to be sent

The following is an example of a COBOL/400 WRITE statement that sends one data record.

```

01 DATA-RECORD.
03 RECORD-LENGTH PIC 9(4).
03 THE-RECORD PIC X(256).
.
.
WRITE TRANSACTION-RECORD FROM DATA-RECORD,
  FORMAT IS '$$SENDNI', TERMINAL IS ICF-PGMDEV.

```

Figure 7-6 on page 7-6 is an example of an RPG/400 output specification to send one data record.

The following is an example of a C/400 write statement that sends one data record.

invited-program-devices operation to receive data from the remote system.

You do not need to issue an invite function before a read operation to receive data. However, if an invite is outstanding for a program device to which a read is issued, the read completes the invite and receives the data.

Refer to Chapter 5 for additional information about the read and read-from-invited-program-devices operations and their relationship to the invite function.

Timer

Your program can use the timer function to set a timer before performing a specified function, such as a read-from-invited-program-devices operation. The timer function specifies an interval of time (in hours, minutes, and seconds) to wait before your program receives a timer-expired (0310) return code.

Use the \$\$TIMER system-supplied format to issue the timer function. The output field for the timer request must be in the following format:

hhmmss

where hh is hours, mm is minutes, and ss is seconds.

The following is a C/400 example that shows how to use \$\$TIMER and set the timer to 30 seconds.

```

| _RFILE *icffptr;          /* Pointer to the ICF file */
| :
| icffptr = _Ropen("ICFFILE","ab+ indicators=y riofb=y");
| :
| _Rformat(icffptr, "$$TIMER"); /* Set timer format */
| _Rpgmdev(icffptr, "CM1");    /* Set default device to CM1 */
| _Rwrite("000030", 6);      /* Issue timer function */
| _Rreadindv(icffptr, &record, sizeof(record), __DFT);
| /* Issue RFI */
|
| /* See if the timer ended by checking 0310 return code */
| if (strncmp(_Maj_Min_rc.major_rc, "03", 2) == 0 &&
|     strncmp(_Maj_Min_rc.minor_rc, "10", 2) == 0)
|     timer_exp();          /* Timer ended, call timer_exp */
|                             /* routine to handle the time out */

```

The following is a COBOL/400 example that shows how to use \$\$TIMER and set the timer to

30 seconds. A read-from-invited-program-devices operation is used to receive the data. The return code must be checked for the timer-expired return code.

```

01 TIMER PIC X(6) VALUE '000030'.
.
.
WRITE TRANSACTION-RECORD FROM TIMER,
    FORMAT IS '$$TIMER', TERMINAL IS ICF-PGMDEV.
.
.
READ TRANSACTION-FILE,
    IF RETURN-CODE EQUAL '0310',
    THEN
        GO TO TIMER-EXPIRED.

```

Your program continues to run, and all operations and functions are valid during the time interval. Your program must issue a read-from-invited-program-devices operation some time after it has issued the timer function, so it can accept the timer-expired return code.

Only one time interval can be maintained for your program. If a previous timer function has been issued and the timer has not yet ended, the old time interval is replaced by the new interval.

The timer function can be used to vary the maximum amount of time that a read-from-invited-program-devices operation will wait for a response. When the time interval set by the TIMER keyword is in effect, the value specified for the WAITRCD parameter on the CRTICFF command is ignored.

You can use the timer function to retry other operations that may not be successful, possibly because of a temporary lack of resources (for example, during an acquire operation). To do this, issue the timer function, and then perform read-from-invited-program-devices operations until the timer ends. (The read-from-invited-program-devices operation allows the program to continue receiving input from other invited program devices while waiting for the timer.)

Refer to Chapter 5 for additional information on the read-from-invited-program-devices operation and its relationship to the timer function.

Figure 7-7 on page 7-8 is an example of using RPG/400 programming language to enter the value on the output specifications to set the timer for 30 seconds.

Program _____ Card Electro Number _____
 Programmer _____ Date _____ Keying Instruction _____ Graphic Key _____
 Page 1 of 2 Program Identification 75 76 77 78 79 80

Line	Form Type	Filename or Record Name	Type (N/D/V/E)		Space		Skip		Output Indicators			Field Name or EXCPT Name	Edit Codes B/A/C/I-S/R	End Position in Output Record R/B/L/R	Constant or Edit Word																																
			D	R	Before	After	Before	After	Not	And	And				Commas	Zero Balances to Print	No Sign	CR	-	x - Remove Plus Sign	Y - Date Field Edit	Z - Zero Suppress	S - User Defined																								
3																1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24								
0 1																																															
0 2																																															
0 3		ICFFILE																																													
0 4																																															
0 5																																															
0 6																																															
0 7																																															

RSL5101-1

Figure 7-7. Timer RPG/400 Output Specification

Problem Notification

The following are the system-supplied formats that may be used to indicate that some type of error has occurred.

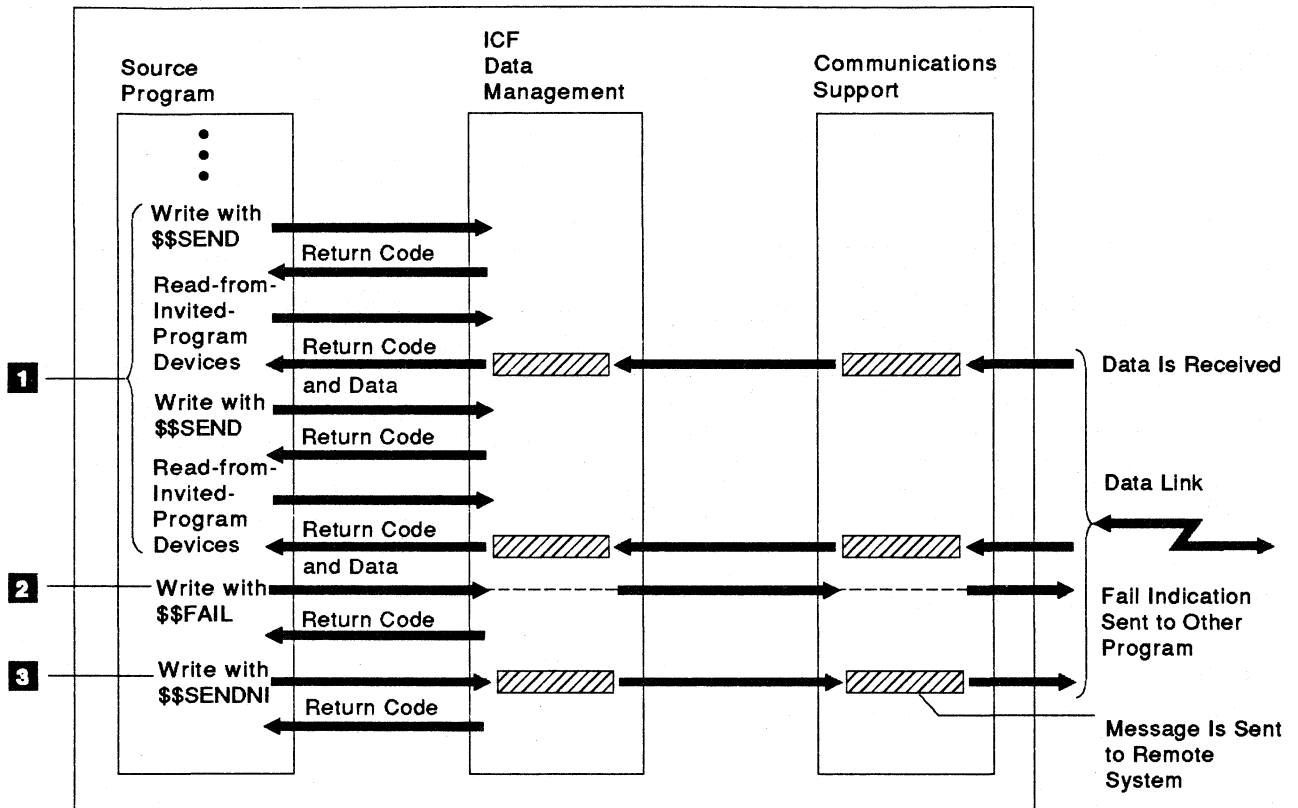
Fail

Use the fail to indicate that your program detected an abnormal condition while sending or receiving data. Use the \$\$FAIL system-supplied format to issue a fail. No data can be sent with the fail.

Figure 7-8 on page 7-9 shows how to use the \$\$FAIL communications format when your program is receiving data and detects an error.

- 1 Your program is receiving data from the remote program.
- 2 While receiving data, your program determines that it must send a fail indication to the other program. A write with the \$\$FAIL communications format is used to send the fail indication.
- 3 A message or data is then sent (write operation) to tell the other program why you sent the fail.

AS/400 System



RSL9672-2

Figure 7-8. Using \$\$FAIL to Send an Error Signal

No output fields are associated with the \$\$FAIL communications format.

The following is a C/400 write statement example that sends a fail indication.

```

| _RFILE *icffptr;          /* Pointer to the ICF file */
|
| :
| icffptr = _Ropen("ICFFILE","ab+ indicators=y riofb=y");
|
| :
| _Rformat(icffptr, "$$FAIL"); /* Set fail format */
| _Rpgmdev(icffptr, "CM1"); /* Set default device to CM1 */
|
| _Rwrite(icffptr, NULL, 0); /* Send the fail */

```

The following is a COBOL/400 WRITE statement example that sends a fail indication.

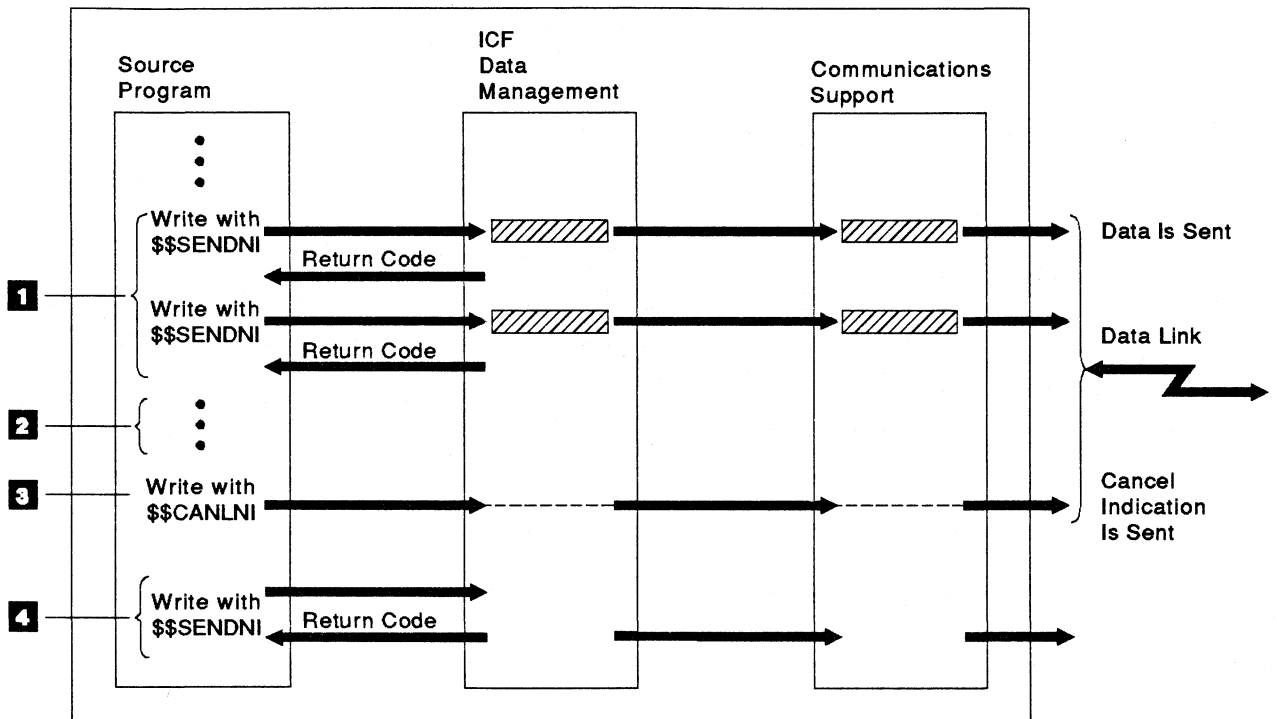
```

WRITE TRANSACTION-RECORD,
    FORMAT IS '$$FAIL', TERMINAL IS ICF-PGMDEV.

```

Figure 7-9 on page 7-10 is an example of an RPG/400 output specification to send a fail indication.

AS/400 System



RSL5673-3

Figure 7-10. Using \$\$\$CANL to Send an Error Indication

- 1 Your program is sending data to the remote system.
- 2 Your program checks the data and determines that something is wrong with it.
- 3 A write operation with the \$\$\$CANLNI communications format is used to tell the remote system to discard the data you have sent.
- 4 Your program can send a message indicating the problem, send the data again, receive more data, or end the transaction.

No output fields are associated with the \$\$\$CANLNI and \$\$\$CANL communications formats.

- | The following is a C/400 write statement example that cancels the current chain of records.

```

_RFILE *icffptr;          /* Pointer to the ICF file */
:
icffptr = _Ropen("ICFFILE","ab+ indicators=y riofb=y");
:
_Rformat(icffptr, " $$$CANLNI"); /* Set cancel w/no invite format */
_Rpgmdev(icffptr, "CM1");      /* Set default device to CM1 */
|Rwrite(icffptr, NULL, 0);    /* Send the cancel */

```

The following is a COBOL/400 WRITE statement example that cancels the current chain of records.

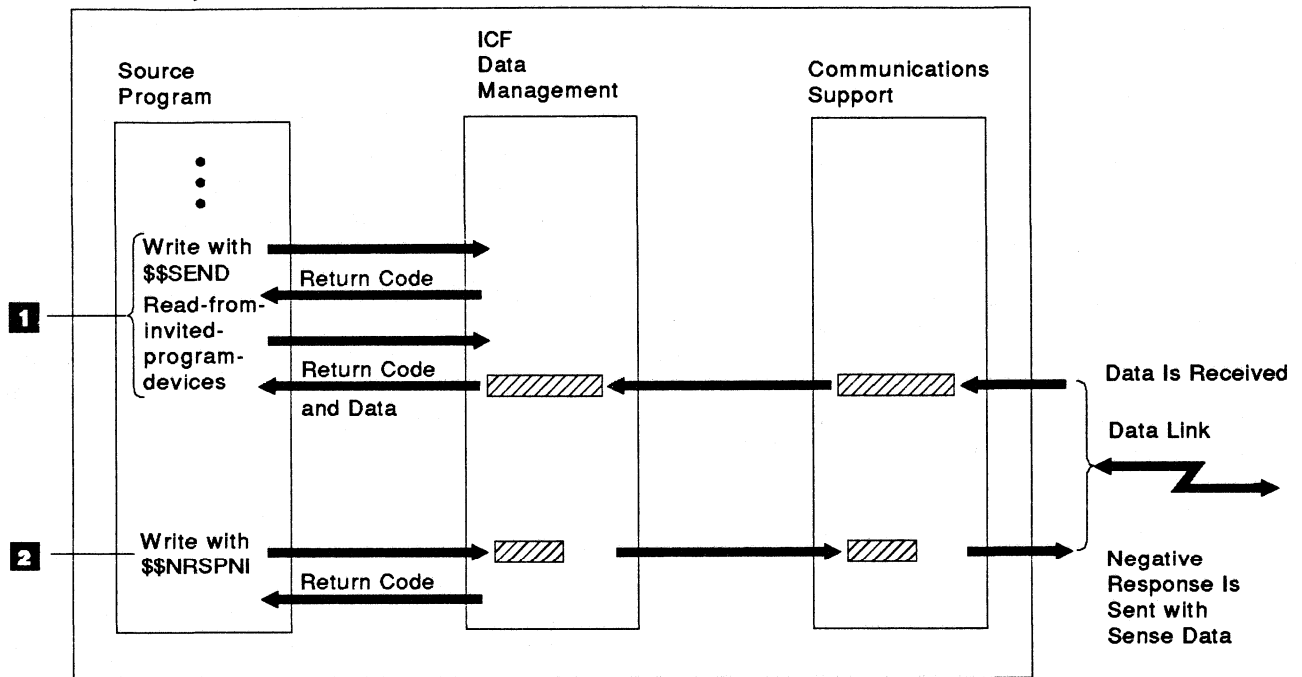
```

WRITE TRANSACTION-RECORD
  FORMAT IS ' $$$CANLNI', TERMINAL IS ICF-PGMDEV.

```

Figure 7-11 on page 7-12 is an example of an RPG/400 output specification to cancel the current chain of records.

AS/400 System



RSL5674-2

Figure 7-12. Using \$\$NRSP to Send an Error Condition

- 1** Your program finds the data it is receiving is not correct.
- 2** The program sends a negative response, by issuing a write with the \$\$NRSPNI communications format, to the remote system including the sense data 08110000. The negative response tells the remote system that the data received is wrong, and the sense data 08110000 asks the remote system to cancel the current group of data records.

The negative-response function requires the fields shown in Figure 7-13 in the output buffer:

Figure 7-13. Sense Data Format

Positions	Description
1	Indicates whether sense data is being sent: 0 or blank indicates that <i>no</i> sense data is being sent; 8 indicates that sense data is being sent.
2 through 9	The sense data sent with the negative response. The first four positions of the sense data must begin with 10xx, 08xx, or 0000. The last four positions are user-defined.

For more information about the allowed sense values, refer to the appropriate communications programming manual for the communications type you are using.

- | The following shows a C/400 write statement that sends a negative (–) response with invite function that includes the sense data 08110000.

```

struct {
    char length;
    char data??(8??);
} neg_resp_rec = {"8", "08110000"};

:
| _RFILE *icffptr;          /* Pointer to the ICF file */
:
| icffptr = _Ropen("ICFFILE","ab+ indicators=y riofb=y");
:
|
| _RFORMAT(icffptr, "$$NRSP"); /* Set negative response format */
| _Rpgmdev(icffptr, "CM1"); /* Set default device to CM1 */
:
| /* Send negative response with sense data */
| _Rwrite(icffptr, &neg_resp_rec, sizeof(neg_resp_rec));

```

The following shows a COBOL/400 WRITE statement that sends a negative response with invite function that includes the sense data 08008000.

```

01 NEG-RESP-REC.
03 REC-LEN          PIC X(1) VALUE '8'.
03 RESP-DATA       PIC X(38) VALUE '08008000'.
.
.
WRITE TRANSACTION-RECORD FROM NEG-RESP-REC,
    FORMAT IS '$$NRSP', TERMINAL IS ICF-PGMDEV.

```

Figure 7-14 shows the RPG/400 output specifications you can use to send a negative response that includes the sense data 08110000.

Positive-Response

Use the \$\$POSRSP system-supplied format to send a positive response to the host system when you are using the SNUF protocol. Using this format can increase your application's performance when the host is waiting for a response. Refer to the *SNA Upline Facility Programmer's Guide* for more information.

Request-to-Write

Your program uses the request-to-write function to indicate that it wants to send something to the remote program rather than to continue receiving data.

Use the \$\$RCD system-supplied format to issue the request-to-write function. This format issues the request-to-write with an invite. After you issue the request-to-write, your program must continue to receive data until it receives a return code indicating that the remote system is ready to begin receiving data. The request-to-write function has no additional associated parameters or data.

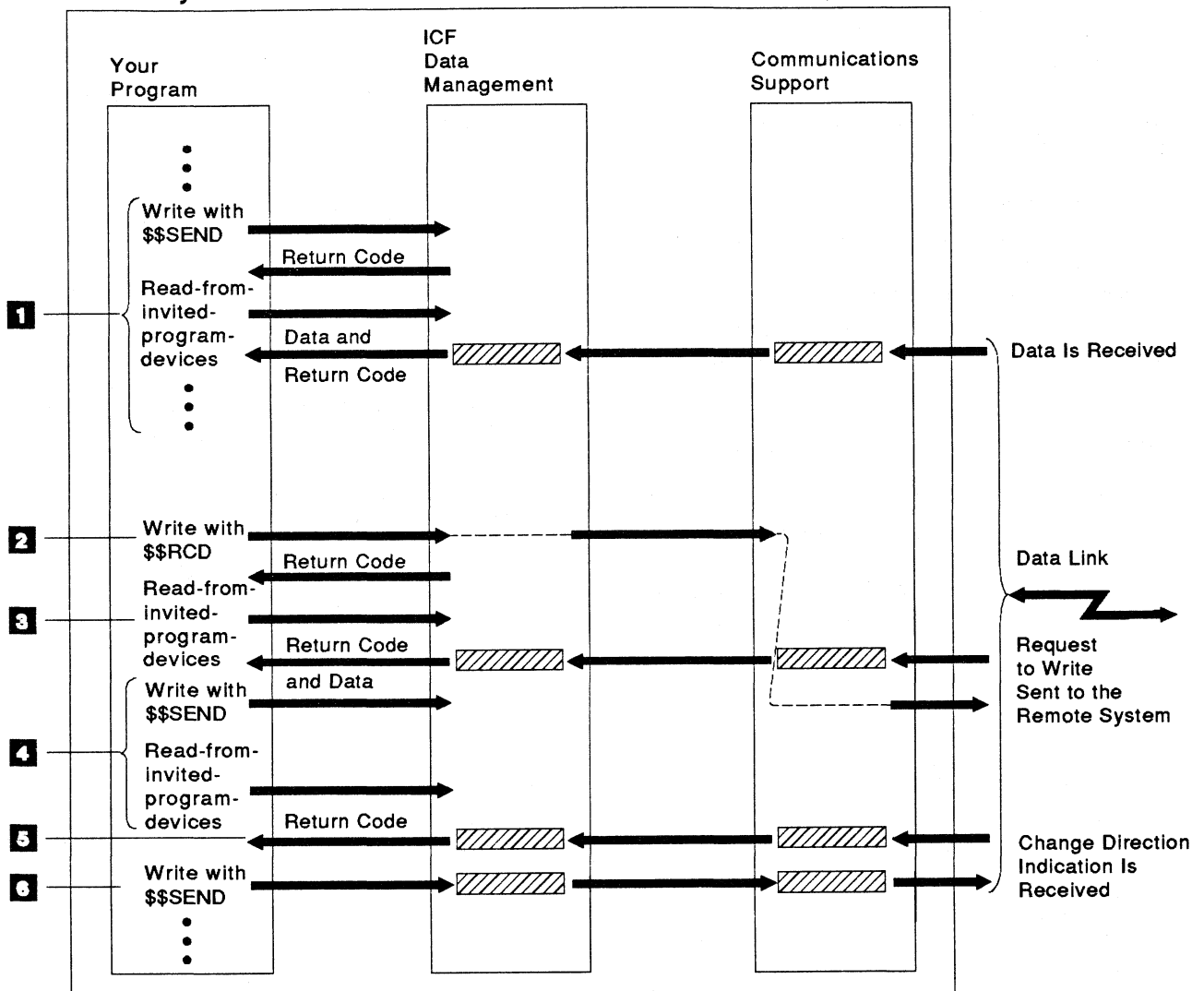
Additional System-Supplied Formats

You can use the following additional system-supplied formats to perform specific tasks when communicating with the remote system.

Program		Keying Instruction	Graphic Key	Card Electro Number	Page 1 2 of	Program Identification 75 76 77 78 79 80										
Line	Form Type	Filename or Record Name	Type (V/D/V/E)	Space	Skip	Output Indicators	Field Name or EXCPT Name	Commas	Zero Balances to Print	No Sign	CR	-	x - Remove Plus Sign	Y - Date Field Edit	Z - Zero Suppress	5 - 9 - User Defined
3	O					And										
4	O					And										
5	O					And										
6	O					And										
7	O					And										
8	O					And										
9	O					And										
10	O					And										
11	O					And										
12	O					And										
13	O					And										
14	O					And										
15	O					And										
16	O					And										
17	O					And										
18	O					And										
19	O					And										
20	O					And										
21	O					And										
22	O					And										
23	O					And										
24	O					And										
25	O					And										
26	O					And										
27	O					And										
28	O					And										
29	O					And										
30	O					And										
31	O					And										
32	O					And										
33	O					And										
34	O					And										
35	O					And										
36	O					And										
37	O					And										
38	O					And										
39	O					And										
40	O					And										
41	O					And										
42	O					And										
43	O					And										
44	O					And										
45	O					And										
46	O					And										
47	O					And										
48	O					And										
49	O					And										
50	O					And										
51	O					And										
52	O					And										
53	O					And										
54	O					And										
55	O					And										
56	O					And										
57	O					And										
58	O					And										
59	O					And										
60	O					And										
61	O					And										
62	O					And										
63	O					And										
64	O					And										
65	O					And										
66	O					And										
67	O					And										
68	O					And										
69	O					And										
70	O					And										
71	O					And										
72	O					And										
73	O					And										
74	O					And										

Figure 7-14. Negative-Response RPG/400 Output Specifications

AS/400 System



RSL9676-2

Figure 7-15. Using \$\$RCD to Request Write

Figure 7-15 shows how to use the \$\$RCD communications format to request permission to send.

- 1** Your program is receiving data from the remote system. The first program processes the data received, then receives data again.
- 2** At some time while data is being received, your program determines that it needs to send a message to the remote system. A write with the \$\$RCD communications format is used to ask the remote system to stop sending so your program can send the message.
- 3** After issuing the request-to-write, your program must continue receiving data until it gets a return code indicating that the remote system is ready to receive. To continue receiving, your program issues another read-from-invited-program-devices operation.
- 4** Another invite function and another read-from-invited-program-devices operation are issued to continue receiving data.
- 5** When the remote system is ready to receive, it sends one more data record with a change-direction indication. The record says the remote system is now ready to receive data or, as in this example, a message.

6 A write with the \$\$\$SEND communications format is used to send the message to the remote system and ask the remote system to continue sending data.

No output fields are associated with the \$\$RCDD communications format.

When your program receives a request-to-write indication from the remote system, a code is set in the input/output (I/O) feedback communications-dependent section. Refer to Figure C-5 on page C-5 for more information about where this field is in the I/O feedback area.

The code indicates the following conditions:

- 0** Continue sending as normal.
- 1** A request-to-write has been received.

For more specific information on what this code means for the communications type you are

using, refer to the appropriate communications programmer's guide.

The following is an example of a C/400 write statement to request the remote system to stop sending data.

```

_RFILE *icffptr;          /* Pointer to the ICF file */
:
icffptr = _Ropen("ICFFILE","ab+ indicators=y riofb=y");
:
:
Rformat(icffptr, "$$RCDD"); /* Set request-to-write format */
Rpgmdev(icffptr, "CM1");   /* Set default device to CM1 */
:
Rwrite(icffptr, NULL, 0);  /* Send the request-to-write */

```

The following is an example of a COBOL/400 WRITE statement to request the remote system to stop sending data.

```

WRITE TRANSACTION-RECORD,
  FORMAT IS '$$RCDD', TERMINAL IS ICF-PGMDEV.

```

Figure 7-16 is an example of a RPG/400 output specification to request that the remote system stop sending data.

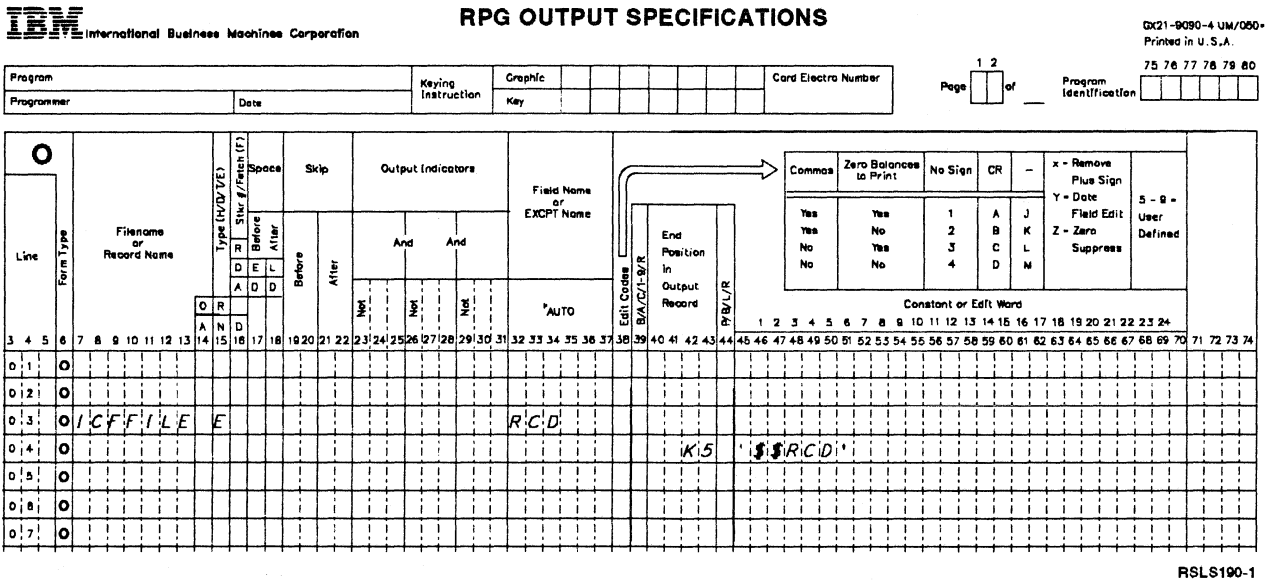
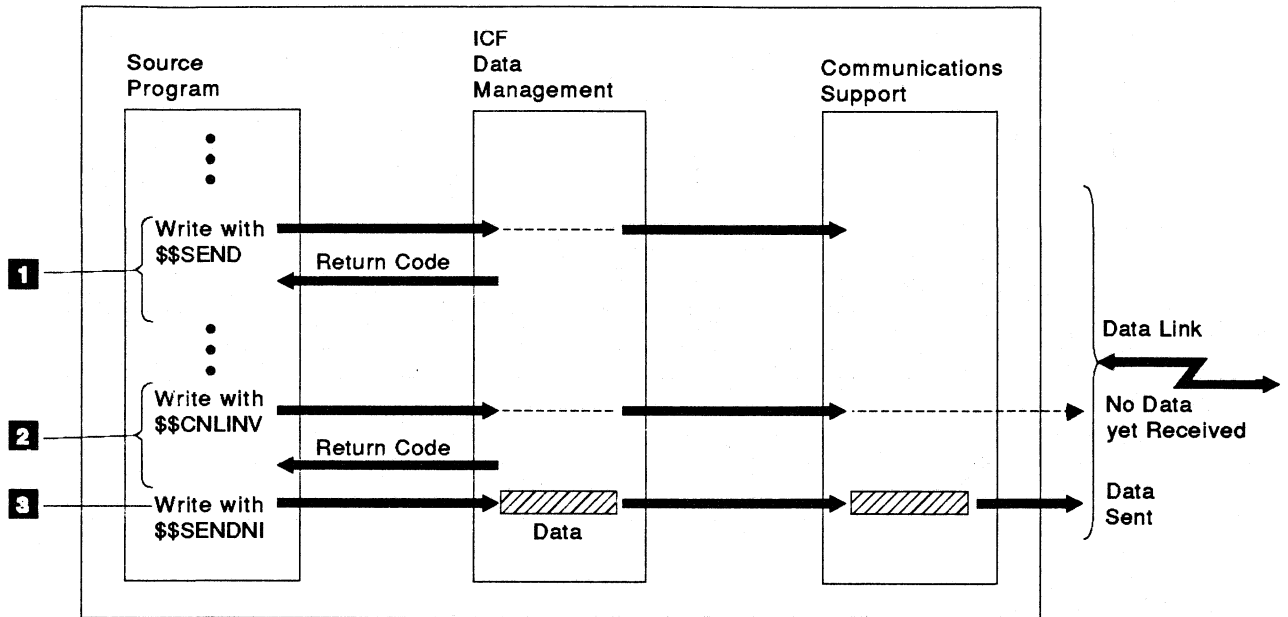


Figure 7-16. Request-to-Write RPG/400 Output Specification

AS/400 System



RSL5675-2

Figure 7-17. Using \$\$CNLINV to Cancel an Invite

Cancel-Invite

Your program uses the cancel-invite function to cancel any valid invite for which no data has yet been received. Use the \$\$CNLINV system-supplied format to issue the cancel-invite function.

Figure 7-17 shows how to use the \$\$CNLINV communications format to issue the cancel-invite function.

- 1** Your program issues an invite to receive data from the remote program, then continues processing.
- 2** Your program can cancel the invite issued previously using the cancel-invite function (issuing a write operation with the \$\$CNLINV communications format). Your program must check the return code it receives to determine if the invite was cancelled.
- 3** If a successful return code is received, your program can send data.

No output fields are associated with the \$\$CNLINV communications format.

The following is an example C/400 write statement that issues a cancel-invite function to a program device that has not received input.

```

| _RFILE *icffptr;          /* Pointer to the ICF file */
|
| :
| icffptr = _Ropen("ICFFILE","ab+ indicators=y riofb=y");
|
| :
| _Rformat(icffptr, "$$CNLINV"); /* Set cancel-invite format */
| _Rpgmdev(icffptr, "CM1"); /* Set default device to CM1 */
|
| _Rwrite(icffptr, NULL, 0); /* Issue the cancel-invite */

```

The following is an example COBOL/400 WRITE statement that issues a cancel-invite function to a program device that has not received input.

```

WRITE TRANSACTION-RECORD
  FORMAT IS '$$CNLINV', TERMINAL IS ICF-PGMDEV.

```

Figure 7-18 on page 7-18 is an example RPG/400 output specification to issue a cancel-invite to a program device that has not received input.

Program				Keying Instruction				Graphic				Card Electro Number				Page 1 of 2				Program Identification			
Programmer				Date				Key												75 76 77 78 79 80			

Line	Form Type	Filename or Record Name	Type (M/D/VE)				Space		Skip		Output Indicators			Field Name or EXCPT Name		Edit Codes B/A/C/I-9/R	Commas				Zero Balances to Print				No Sign				CR				-				x - Remove Plus Sign				Y - Date Field Edit				Z - Zero Suppress				5 - 9 - User Defined			
			A	D	D	D	Before	After	Before	After	Not	Not	Not	1	2		3	4	1	2	3	4	A	B	C	D	J	K	L	M	x	y	z	5	6	7	8	9														
01	O																																																			
03	O	ICFFILE												CNLIWV																																						
04	O																																																			
05	O																																																			
06	O																																																			
07	O																																																			

RSLS189-1

Figure 7-18. Cancel-Invite RPG/400 Output Specifications

Ending a Communications Transaction

A communications transaction can be ended by your program or by the program at the remote system. Your job and the remote system your system is communicating with determine the program that ends the transaction.

Communications with the remote program end when your program ends the transaction. However, the session may still exist if your program started the session. If the session still exists, you can end the session or you may be able to start another program at the remote system.

Detach

Use the detach function to end the transaction. The detach function explicitly informs the remote program that your program is done sending, and ends the transaction.

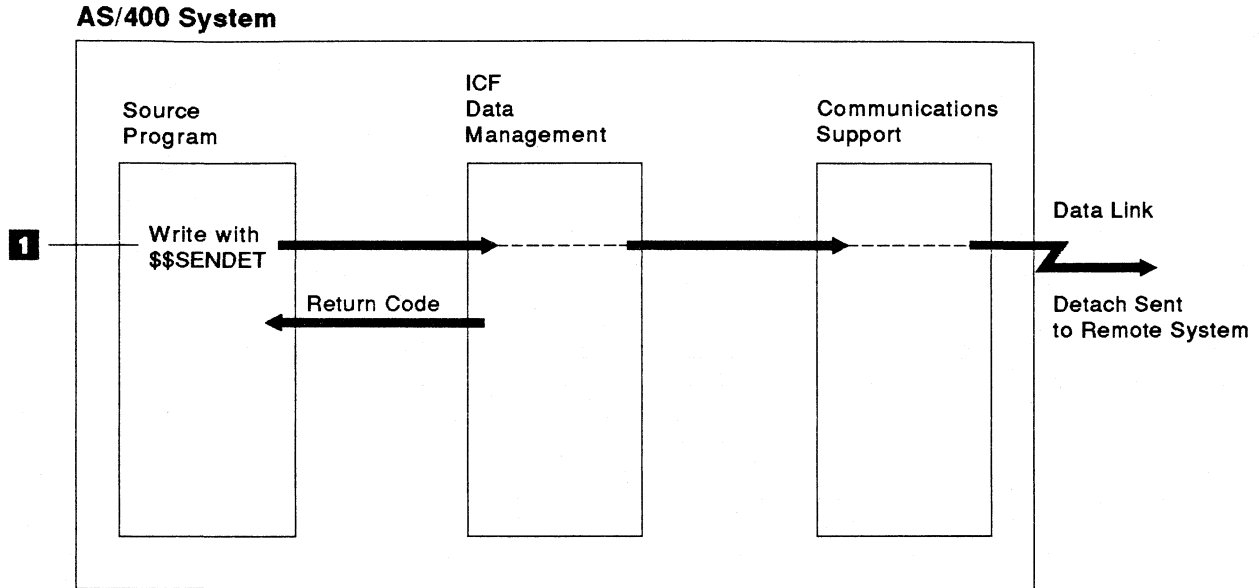
You can use one of the following system-

supplied formats to perform a detach function (in combination with other functions):

- **Evoke with Detach (\$\$EVOKET).** Starts the specified program on the remote system and ends the transaction without allowing the target program to communicate in return.
- **Send with Detach (\$\$SENDET).** Sends a data record to the remote program and ends the transaction. Note that not all communications types support sending a data record and the detach function on the same operation. Refer to the appropriate communications manual to determine if you can send data with the detach function.

You can use the \$\$SENDET communications format with an output length of zero to issue a detach function by itself.

Figure 7-19 on page 7-19 shows how to use the \$\$SENDET communications format to end the transaction.



RSL5678-2

Figure 7-19. Ending the Communications Transaction

- 1 Your program issues the detach function, by using a write with the \$\$\$SENDT system-supplied format, to tell the remote system that your program ended the communications transaction.

Refer to "Starting a Program on the Remote System" on page 7-2 and "Sending Data" on page 7-4 for information about the output format of these functions.

- The following is an example of a C/400 write statement to issue a detach.

```

struct {
    char data_length??(4??);
    char data??(80??);
} data_rec;

:
_RFILE *icffptr;          /* Pointer to the ICF file */
:
icffptr = _Ropen("ICFFILE","ab+ indicators=y riofb=y");
:
_Rformat(icffptr, "$$$SENDT"); /* Set write-with-detach format */
_Rpgmdev(icffptr, "CM1");     /* Set default device to CM1 */

strncpy(data_rec.data_length, "0080", 4); /* Set record length */
_Rwrite(icffptr, &data_rec, sizeof(data_rec)); /* Send detach */

```

- The following is an example of a COBOL/400 WRITE statement to issue a detach.

```

01 DATA_RECORD.
03 RECORD-LENGTH          PIC 9(4).
03 THE-RECORD             PIC X(256).
.
.
WRITE TRANSACTION-RECORD FROM DATA-RECORD,
    FORMAT IS '$$$SENDT', TERMINAL IS ICF-PGMDEV.

```

If a detach function is issued by a target program, the session is ended as well as the transaction.

Figure 7-20 on page 7-20 is an example of an RPG/400 output specification to issue a detach.

Program	Keying Instruction	Graphic Key	Card Electro Number	Page 1 of 2	Program Identification
Programmer	Date				75 76 77 78 79 80

Line	Form Type	Filename or Record Name	Type (IND/VE)		Space	Skip	Output Indicators			Field Name or EXCPT Name	Edit Codes B/A/C/D/E/R	End Position in Output Record	Constant or Edit Word																								
			Before	After			And	And	Commas				Zero Balances to Print	No Sign	CR	-	x - Remove Plus Sign	Y - Date Field Edit	Z - Zero Suppress	5 - g - User Defined																	
01	O	ICFFILE	E							SEND			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
02	O											KB	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
03	O											4	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
04	O									LISTREC		BA	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
05	O																																				
06	O																																				
07	O																																				

RSL5102-1

Figure 7-20. Detach RPG/400 Output Specification

Ending the Communications Session

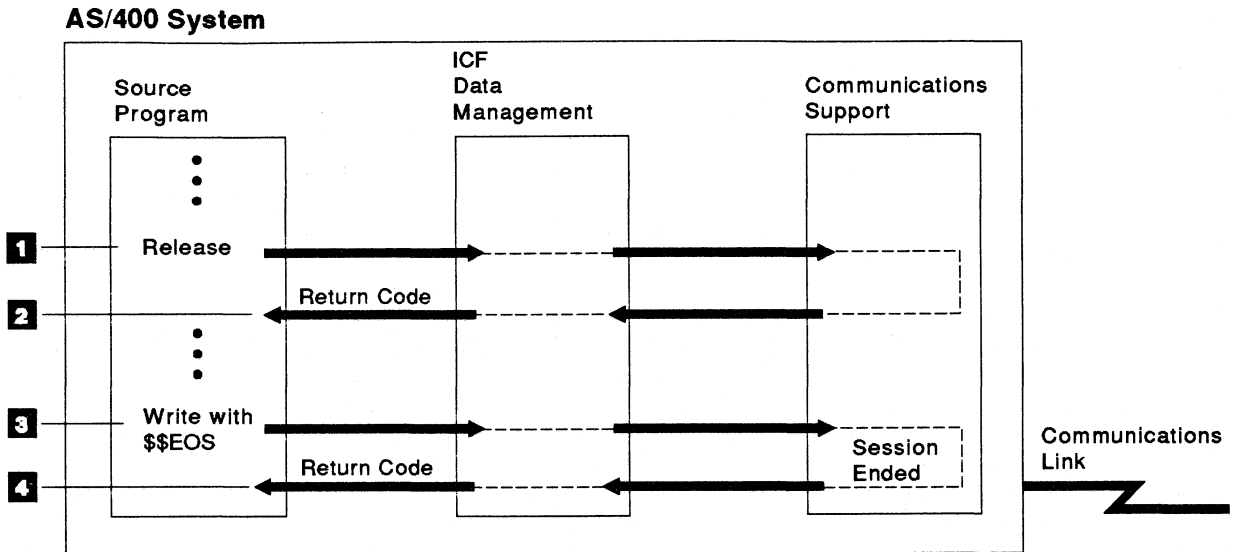
How the communications session is ended depends on whether your program or the remote system started the session.

If your program started the session (source program), your program must end the session using either the release operation or the end-of-session function. You should primarily use the release operation. Use the end-of-session function only when you want to force the session to end.

The release operation ends the session only if all processing is complete. The end-of-session function *always* ends the session. The only possible return codes from end-of-session are 0000 or 830B (program device not acquired).

End of Session

Use the \$\$EOS system-supplied format to issue the end-of-session function. No data can be sent with the end-of-session function. Figure 7-21 on page 7-21 shows how you can end the session by using the release operation and the end-of-session function.



RSL5677-2

Figure 7-21. Using the Release Operation and End-of-Session Function

- 1 Your program uses the release operation to end the current communications session.
- 2 The return code tells your program whether the session was ended or an error occurred while trying to end the session. If, for example, all transactions have not ended when the release operation is issued, an error occurs and the session is not ended.
- 3 If an error occurs and normal recovery is not possible, your program can use the end-of-session function to end the session.
- 4 The end-of-session function always ends the session.

If the target program ends the transaction by the detach function, the session is ended implicitly. If the source program ends the transaction, the target program must issue an end-of-session function or go to the end of the job to end the session.

No output fields are associated with the \$\$EOS communications format.

The following is an example of a C/400 write statement specifying the \$\$EOS format.

```

_RFILE *icffptr;          /* Pointer to the ICF file */
:
icffptr = _Ropen("ICFFILE","ab+ indicators=y riob=y");
:
_Rformat(icffptr, "$$EOS"); /* Set end-of-session format */
_Rpgmdev(icffptr, "CM1");  /* Set default device to CM1 */
_Rwrite(icffptr, NULL, 0); /* Send an end-of-session */

```

The following is an example of a COBOL/400 WRITE statement specifying the \$\$EOS format.

```

WRITE TRANSACTION-RECORD,
    FORMAT IS '$$EOS', TERMINAL IS ICF-PGMDEV.

```

Figure 7-22 on page 7-22 is an example of an RPG/400 output specification to issue the end-of-session function using the \$\$EOS format.

Figure 7-23 (Page 2 of 2). System-Supplied Format Support

System-Supplied Formats	APPC	SNUF	BSCCL	Asynchronous	Intrasystem	Finance	Retail
\$\$SENDFM		X			X	X	X
\$\$SENDNF		X		X	X	X	X
\$\$SENDNI	X	X	X	X	X	X	X
\$\$TIMER	X	X	X	X	X	X	X

¹ These keywords are not valid for the 3694 controller. Refer to the *Finance Communications Programmer's Guide* for more details.

Mapping System-Supplied Formats to DDS Keywords

Figure 7-24 maps system-supplied formats to DDS keywords.

Figure 7-24. Mapping of System-Supplied Formats to DDS Keywords

System-Supplied Formats	DDS Keywords
\$\$CANL	CANCEL with INVITE
\$\$CANLNI	CANCEL
\$\$CNLINV	CNLINVITE
\$\$EOS	EOS
\$\$EVOK	EVOKE, SECURITY, and INVITE
\$\$EVOKET	EVOKE, SECURITY, and DETACH
\$\$EVOKNI	EVOKE and SECURITY
\$\$FAIL	FAIL
\$\$NRSP	NEGRSP with INVITE
\$\$NRSPNI	NEGRSP
\$\$POSRSP	RSPCONFIRM
\$\$RCD	RQSWRT with INVITE
\$\$SEND	INVITE
\$\$SENDE	ENDGRP
\$\$SENDET	DETACH
\$\$SENDFM	FMH with INVITE
\$\$SENDNF	FMH
\$\$SENDNI	No DDS keywords
\$\$TIMER	TIMER

Chapter 8. Programming Considerations

This chapter presents general communications programming considerations related to the ICF file. Programming considerations specific to a communications type are not discussed. Refer to the appropriate communications programming manual for more information on the programming considerations for a particular communications type.

Return Codes

Return codes are used by the communications application program to determine the program state. Program states are receive, send, or exception. The program checks the return codes and completes the action required by the contents of the return codes.

The meaning of the major ICF return codes and some examples of minor return codes follow. These definitions help you determine the return codes you need to check in your program. For a complete list of return codes, see Appendix B.

Major Codes

All major return codes that represent normal conditions have values 00xx, 02xx, and 03xx. Major return codes that represent input/output (I/O) exceptions have values of 04xx and 34xx. Major return codes that represent *error* conditions have values 08xx, 11xx, and 80xx through 83xx. This division lets you quickly compare codes and determine the type of action required.

The main groups of major return codes are:

- Operation was successfully completed (00xx, 02xx).
- Successful operation. No data was received, but some control information may have been received (03xx).
- Output exception occurred (04xx).
- Miscellaneous program errors occurred (08xx and 11xx).
- Input exception occurred (34xx).
- Nonrecoverable error occurred. The session has been ended and the underlying commu-

nications support may no longer be active (80xx).

- Nonrecoverable session error occurred. Session has been ended but the underlying communications support is still active (81xx).
- Open or acquire operation failed. Session was not started but recovery might be possible (82xx).
- Session error occurred. Recovery might be possible (83xx).

Minor Codes

The minor part of a return code identifies the specific condition within the general condition that is identified by the major part of the code. Some examples of the minor codes are:

- A detach was received (xx08), a detach was received with a system message (xx18), or a detach was received with a confirm (xx1C).
- An invalid evoke function was issued (xx29).

ICF Environment Considerations

You must understand the following considerations before you write programs for ICF communications:

- Your program should check the major/minor return code after every operation to determine the success or failure of the operation.
- Information in the open and I/O feedback areas can be useful to your program. Refer to Chapter 5 for information about these areas and Appendix C for a summary of the fields available in these areas.
- A target program can communicate with the source program by acquiring the program device whose remote location name is *REQUESTER.
- If a target program never acquires a program device for the requesting program device, a diagnostic message is written to the job log when the target job completes. The message indicates that the program ended with an active connection to the session. The message is normal if there is

no need for the target program to communicate with the source program. If the target program must communicate with the source program, this message indicates a possible logic problem in your program.

- For most communications types, the first I/O operation following the acquire by the source program must include an evoke function. This operation starts the target program on the remote system with which the source program is to communicate. You can start the target program with program initialization parameters specified as part of the evoke function.
- When a single program is written to function as either a source or a target program, the program may need to determine its role (source or target). A suggested method of making this analysis for a program is to define a CALL parameter that contains one value when the program is started by the remote system and another value when the program is started by the local user or program.
- A target program cannot start error recovery. If a permanent session error occurs, the target program should finish any needed processing and end the program. The source program is responsible for reestablishing the session and transaction. See Appendix B for more information about communications error recovery.

Open or Acquire Considerations

The following open or acquire considerations apply when you write programs for ICF communications:

- If an ICF file open is a subsequent open of a shared file, the program is attached to the already open file. The state and attributes of the file do not change. Refer to the *Data Management Guide* for more information on shared file processing.
- Your program can establish more than one session. These sessions can be to the same remote system (if the remote system supports multiple sessions) or to different remote systems. The program device names are used to distinguish between sessions within your program.

- A program device can either be acquired automatically when the ICF file is opened (ACQPGMDEV = program device) or explicitly with the acquire operation. Only one program device can be acquired as part of the open operation. If a program is using multiple sessions, all of the program devices, except the first one, must be explicitly acquired.

Note: With a FORTRAN/400 program, a separate ICF file must be used for each ICF session. FORTRAN/400 programs can implicitly acquire a program device only when the ICF file is opened (ACQPGMDEV = program device).

- If a target program acquires a program device other than a program device associated with a remote location with the name of *REQUESTER, a new session is allocated and a logical link to the source program is not established. No error is indicated because a target program on one session can also be a source program on another session.

Output Considerations

The following output considerations apply when you write programs for ICF communications:

- Your program should check the major/minor return code after every output operation to determine if the remote system wants to send data. You can also use a field in the I/O feedback to determine this information. See Appendix C for a summary chart of the I/O feedback area.
- When the program is communicating with multiple sessions, the appropriate program device name must be specified on the write statement in the control area (depending on the language used) before issuing the output operation. Refer to examples in Chapter 9 through Chapter 11.
- The output length of the operation is determined by the specified record format. If you use user-defined formats, the record format length, as determined by the record definition in data description specifications (DDS), is the output length. You can vary this output length at run time by using the VARLEN DDS keyword. If you use system-

supplied formats and data is allowed as part of the function, the output length is specified as part of the record format.

- If an output operation is issued with a zero record length, ICF assumes that the needed functions are only the functions indicated by the operation specified, and the data is not placed in the output buffer. For example, if a zero-length write operation is issued with a user-defined format with the INVITE keyword in effect, the program device is invited, but no data is sent to the remote system.
- Multiple output operations can be done with a single write operation. For example, if a write operation is issued with a user-defined format with the FMH and INVITE keywords in effect, or with the system-supplied \$\$\$SENDFM format, data is sent to the remote system with FMH information and the program device is invited. Refer to Chapter 6 to determine the processing sequence of the DDS keywords.
- If your program issues an output operation with an output length greater than the length supported for the session, the operation completes with an 831F return code. The maximum output length supported is determined by the communications type you are using, the record length specified in configuration, or the record length specified on the Add Intersystem Communications Function Device Entry (ADDICFDEVE) or Override Intersystem Communications Function Device Entry (OVRICFDEVE) command.
- If your program issues an output operation while a session is in receive state, the operation may complete with an output exception (a major/minor return code of 0412). If the operation completes with an output exception, the data is not sent to the remote system. Your program must issue an input operation to receive the data or system message that is pending.
- If your program issues an output operation to an invited program device and the communications type you are using supports the cancel-invite function, ICF tries to cancel the invite. If the invite cannot be canceled, the output operation completes with a 0412 return code.
- When your program is reading data from a local source, such as a database file, it may

not determine until the next read that it has just read the last record in the file. If this is the case and your program uses a specific indication, such as the end-of-group or detach function, to inform the remote system when the end-of-file indication is reached, then you must ensure that this end-of-file indication is sent with a zero-length record format. If not, then any data in the output buffer from a previous operation is sent to the remote system as user data (for example, the last record may be sent twice).

Input Considerations

The following input considerations apply when you write programs for ICF communications:

- Your program must examine the major/minor return code to determine if the input operation completed with data. A major code of 00 indicates data reception. A major code of 02 indicates that data was received, but the job is being canceled. A major code of 03 indicates that no data was received. Refer to Appendix B for a complete summary of major/minor return codes.
- When the program is communicating with multiple sessions, the appropriate program device name must be specified on the read statement in the control area (depending on the language used) before issuing the input operation. Refer to examples in Chapter 9 through Chapter 11.
- The input length of the operation is determined by the specified record format. If you use user-defined formats, the record format length, as determined by the record definition in DDS, is the input length. If you use the system-supplied QICDMF file, the input length is always 4096.
- When the actual length of the data received is less than the input length, the system pads the remainder of the record with blanks (except for basic conversations in APPC). A field in the I/O feedback area indicates the actual length of the data received from the remote system. Refer to Appendix C for a summary chart of the I/O feedback area.
- For most communication types, when your program issues an input operation that completes with a return code of 0000 or 0300:

- The operation is successful.
- Your program controls the session and can send data.

When the operation completes with a return code of 0001 or 0301:

- The operation is successful.
 - The remote program controls the session and your program should continue issuing input operations.
- Response indicators (RCVTRNRND, RCVDETACH, RCVCONFIRM, RCVNEGRSP, RVCANCEL, RCVFMH, RCVFAIL, and RCVENDGRP) can be received either with data or without data (indicators only). Your program must examine the major/minor return code to determine if the record contains data.
 - For most communications types, if an input operation is issued before an evoke is sent or after a detach function is sent or received, the operation completes with a 8327 return code.
 - When processing input data, consideration should be given to detecting invalid data within a numeric field. The program can detect invalid data by verifying numeric field contents through program logic, or by permitting the system to detect a decimal-data error when the field is used in an arithmetic operation within the program.

The format selection processing used (as determined by the FMTSLT parameter on the ADDICFDEVE or OVRICFDEVE command) determines how the record format name is selected to process incoming data.

If the FMTSLT(*RECID) option is used, ICF returns the record format based on the data received from the remote system. Therefore, the proper record (the one that matches the data received) will be selected.

If the layout of the record format is described within the program, the program logic determines the placement and types of fields within the record. The program is responsible for processing the data according to a specific record type.

Release, End-of-Session, and Close Considerations

The following release, end-of-session, and close considerations apply when you write programs for ICF communications:

Release Considerations

The following are release operation considerations for ICF programs:

- If a target program issues a release operation, the logical connection between the current program and the communications session is ended. The communications session remains intact, and can, by using an acquire operation, be used again in the same job. The session state will be the same as it was when it was released.
- If a source program issues a release operation, the state of the operation is verified, and the release request can be rejected. If the program device is invited, the release operation is rejected with an 832C return code.

If the release operation is successful, the communications session ends.

Note: The FORTRAN/400 language does not support a release operation. Instead, an end-of-session function or a close operation must be issued. Refer to “End-of-Session Considerations” and to “Close Considerations” for more information.

End-of-Session Considerations

The following are end-of-session function considerations for ICF programs:

- An end-of-session function is always successful if a session exists. The system ends the communications session regardless of the state of the session.
- If a target program issues an end-of-session function, the session is ended and cannot be reestablished with an acquire operation.
- A session remains allocated to the target program until an end-of-session function is performed or until the job ends.

- An end-of-session function issued to an unacquired device results in an 830B return code.

Close Considerations

The following are close operation considerations for ICF programs:

- If a close operation is issued to a shared file, the program issuing the close cannot do I/O operations to the file, but other programs that have the file open can still use the file. Refer to the *Data Management Guide* for more information on shared file processing.
- When a close operation is issued to an ICF file, all sessions associated with a source program for the specified file are ended regardless of the state of the session. Depending on the communications type, any buffered data may or may not be sent to the remote system.
- If a session is associated with a requesting program device, the logical connection between the current program and the communications session is ended. The communications session remains intact, and can, by using an acquire operation, be used again in the same job. The session remains intact. The state of the session remains the same when the program device is acquired again.
- Any active transactions associated with the file may be abnormally ended.

Remote Program Start Considerations

Program start requests that are received from a remote system result in an attempt to start the job. The program specified by the program start request runs within a routing step of the job. All jobs on the AS/400 system operate in an environment called a subsystem.

In order to receive program start requests from a remote system and to start a job to run the program specified on the program start request, a subsystem must be defined on the AS/400 system. You can define a new subsystem, or modify an IBM-supplied subsystem such as QBASE or QCMN, to receive and process program start requests.

Refer to the *Work Management Guide* for more information about QBASE and QCMN.

Defining the Environment

Subsystem descriptions are created using the Create Subsystem Description (CRTSBSD) command. The CRTSBSD command is described in the *Work Management Guide*.

The AS/400 system considers a communications device to be another source of work for a subsystem. Therefore, you must define a communications entry in the subsystem description to identify the communications devices and remote locations for which work (program start requests) can be received by the subsystem. Default communications entries are shipped with the system. However, you can change these entries with the following commands:

- Add Communications Entry (ADDCMNE)
- Remove Communications Entry (RMVCMNE)
- Change Communications Entry (CHGCMNE)

You can specify a default user on the communications entry for a subsystem description on the DFTUSR parameter on the ADDCMNE command or CHGCMNE command. Refer to the *Communications Management Guide* for more information on the use of these commands.

Use the Add Routing Entry (ADDRTGE) command to define the routing information for remotely started jobs. You can specify that the program identified in the program start request be used. Or, you can define routing entries that select the program based on the device description name, mode name, or user profile name. Refer to the *Work Management Guide* for more information on the use of this command.

Handling Program Start Requests

When a subsystem receives a program start request from a remote system, it locates a user profile for the job based on one of the following:

- The password or user ID from the program start request (if one was specified)
- The user ID from the DFTUSR parameter of the ADDCMNE command

If the user ID and password are not passed as part of the program start request (for example,

*NONE specified on the SECURITY keyword), the system checks the communications entry in the subsystem handling the program start request to see if it allows a default user. If the entry is not found, the program start request is rejected with a security violation error.

The user profile contains the authorization to the objects and functions the job can reference.

The subsystem also locates a job description for the job. The job description defines job attributes, such as the job output queue and the first library list made. You can specify a job description on the ADDCMNE command or in the user profile.

The program name and the library name are passed as part of the program start request. This information is used along with the subsystem description routing entries to start a routing step and to select the program that starts running on the routing step.

If a program start request is received on the AS/400 system with an unqualified program name, the system uses the library list for the subsystem handling the program start request. The library list for the subsystem consists of the values from the QSYSLIBL and QUSRLIBL system values at the time the subsystem was started.

If a program start request is received on the AS/400 system with a qualified program name, the acceptable format varies by communication type. Most communication types (for example, asynchronous, BSCCEL, and SNUF) separate the library and program as part of the program start request. Intrasystem communications allows the program name to be in the form *library/program*. APPC accepts either *library/program* or the architected form *program.library*.

Note: Program and library names on the AS/400 system are limited to 10 characters. If you are using BSCCEL to start a remote program, the program and library name is limited to 8 characters.

The subsystem searches its routing table to determine the name of the program to be run in the job's routing step. In an interactive or batch environment, the routing data normally selects the program QSYS/QCMD, which processes control language (CL) commands. However,

QCMD does not process data received with program start requests, and should not be selected as a communications target program. For remotely started jobs, the program to be run is commonly specified on the program start request from the remote system.

The subsystem also uses the routing entry to select a class for the job. The job class defines operation attributes, such as operation priority and time slice.

The system sends message CPF1269 to the QSYSOPR message queue if it is unable to start the requested program. The information in the message can help determine and correct the cause of the problem when working with the remote system programmer. Refer to Appendix B for additional information on the message generated for failed program start requests.

When the communications target job is started and the target program begins running, the target job can access any parameters specified on the evoke request as if they were parameters passed on a Call a Program (CALL) command. The target job communicates with the source program by opening an ICF file, acquiring the requesting program device, and issuing I/O requests to read and write data.

If program initialization parameters are passed on an evoke function or program start request, the following points should be noted:

- If multiple program initialization parameters are passed, the system uses commas to separate these parameters. Therefore, do not include commas in your program initialization parameter data.
- If you specify program initialization parameters with the evoke function, each parameter that is sent should be equal in length to the corresponding parameter specified in the target program. If it is longer than the parameter length in the target program, truncation occurs. If it is shorter than the parameter length in the target program, results may occur that cannot be predicted.

The target job runs as a normal batch job, and is subject to all job control commands, such as Display Job (DSPJOB), Work Job (WRKJOB), and End Job (ENDJOB). The job can be transferred

using the Transfer Job (TFRJOB) command, but not the Transfer Batch Job (TFRBCHJOB) command.

Once the transaction ends (when a detach is successfully sent or received), the target program can no longer communicate with the source program. For most communications

types, if the program is started with an evoke-with-detach function and you do an acquire operation to the requesting program device, the acquire fails with a 82A9 return code.

Figure 8-1 shows a sample ICF communications environment.

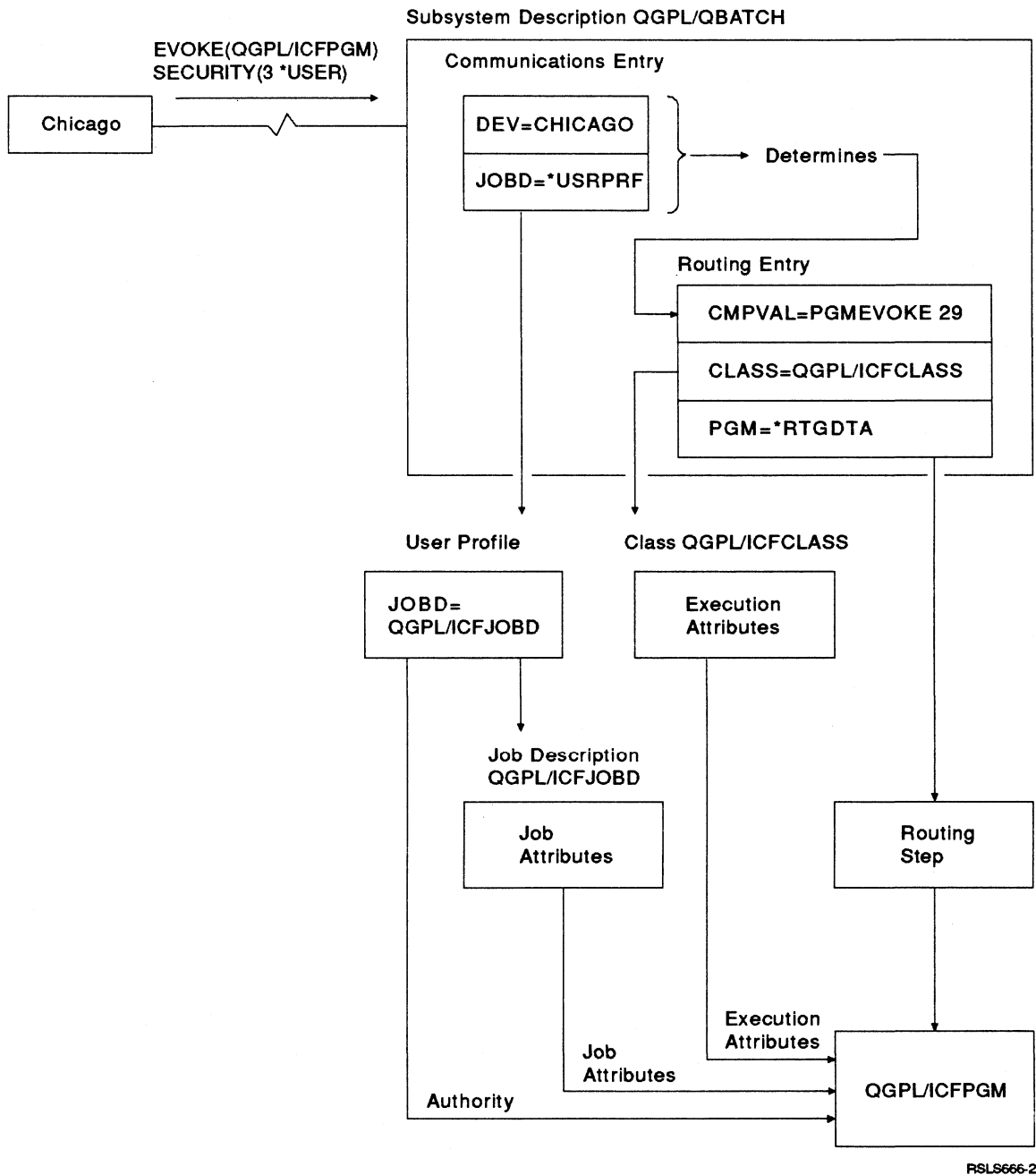


Figure 8-1. Sample ICF Communications Environment

Refer to the *Work Management Guide* for more information on job structures and job processing.

Prestarting Jobs for Program Start Requests

To minimize the time required to carry out a program start request, you can use the prestart job entry to start a job on the AS/400 system *before* the remote program sends a program start request. Because a job is already started and running before the program start request is received, improved program response time results.

You can use prestart jobs for all communications types that support program start request processing: APPC, asynchronous, BSCEL, intra-system, finance, retail, and SNUF communications.

To use prestart jobs, you must define a prestart job entry and a communications entry in the subsystem description. Each prestart job entry contains a program name, library name, user profile, and other attributes, which the subsystem uses to create and manage a pool of prestart jobs.

Both the communications and prestart job entries must be specified in the same subsystem. If a program start request is received on a subsystem that does not have a prestart job entry with the matching program name, the usual processing required to start a communications batch job occurs when the program start request is received. The subsystem attempts to allocate the communications devices that are identified by the communications entries. When a program start request is received, it is sent to the subsystem that has the required communications device allocated. For more information about adding, changing, and removing communications entries in subsystem descriptions, refer to the *Communications Management Guide*.

When a subsystem is initially started, prestart jobs are started based on the information contained in the prestart job entries. Each prestart job entry identifies the program that is to be started, the number of jobs that need to be started with the program, and the user profile under which the jobs will run when it is *not* ser-

ving a program start request. However, when a program start request is received, the subsystem determines if the program name sent on the program start request matches the program name on one of the prestart job entries. If so, the subsystem ensures that the program start request user ID and password are valid and that the user is authorized to use the device and the library and program. The program start request is then attached to a prestart job. The prestart job runs under the user profile specified on the program start request while it is servicing that request.

Commands

Prestart jobs can start at the same time the subsystem is started, or you can use the Start Prestart Jobs (STRPJ) command to start jobs for a prestart job entry in an active subsystem.

The Change Prestart Job (CHGPJ) command allows you to change some of the attributes for a prestart job based on either the attributes of the job description for a prestart job entry or the attributes of the job description defined in the user profile of a program start request.

The End Prestart Job (ENDPJ) command allows you to end all jobs for a prestart job entry in an active subsystem.

The following commands should be used when working with prestart job entries in the subsystem description:

- The Add Prestart Job Entry (ADDPJE) command adds a prestart job entry to the specified subsystem description.
- The Change Prestart Job Entry (CHGPJE) command changes a prestart job entry in the specified subsystem description.
- The Remove Prestart Job Entry (RMVPJE) command removes a prestart job entry from the specified subsystem description.
- The Display Active Prestart Jobs (DSPACTPJ) command displays the run time statistics and performance information for prestart jobs associated with a prestart job entry in an active subsystem.

For more information about the parameters and attributes associated with these commands, refer to the *CL Reference* manual.

Application Considerations

Certain programming changes need to be made to the prestart job program to allow it to be started by and communicate with the remote program. The following points must be taken into account when writing prestart job applications:

- A prestart job program should do as much work as possible, for example, allocating objects and opening database files, before attempting to acquire a requesting program device. Once a prestart job is started, this initial processing is done before the acquire of a requesting program device. The acquire operation causes the job to wait until a program start request is received. When a program start request is received, the program then continues with the acquire operation.
- When a prestart job program is done servicing a program start request, it must do an end-of-session function followed by the acquire of a requesting program device. This is the only way the prestart job makes itself available for the next program start request. If a release operation is performed instead of an end-of-session function, the acquire of the requesting program device does not cause the program to wait, and the program device continues to run on the current session.
- Because a job is already started and running before a program start request with program initialization parameters is received, the subsystem stores these parameters in the program initialization parameter data area for the prestart job to which the program start request attaches. After the acquire of the requesting program device, you must use the Retrieve Data Area (RTVDTAARA) command (specifying *PDA as the data area) or the high level language Retrieve Data Area operation (for example, COBOL ACCEPT) to retrieve the program initialization parameters (if any) passed on the program start request.

Note: A maximum length of 2000 bytes is allowed for program initialization parameters passed on a program start request for a prestart job.

- Only resources that are used specifically for a transaction should be deallocated. Any resource that is commonly used for most transactions performed by the prestart job program should remain allocated while the job is waiting for the next request to be received.
- When a program start request attaches to a prestart job, none of the attributes associated with the user profile on the program start request are used. To change the attributes for a job to those of the job description on the user profile specified on the program start request, use the CHGPJ command.
- Your application should check for an 8209 return code after completion of the acquire to the requesting program device to determine if the prestarted job is being canceled.

See Figure 8-2 on page 8-10 for a sample prestart job program. For information about sharing database files in the same job and across jobs, see the *Database Guide*.

Security Considerations for Prestart Jobs

When a prestart job is initially started, authority checking for a prestart job entry user profile is performed on every object that is needed to run the job. When a program start request attaches to a prestart job, however, it runs under the user profile specified on the program start request. Before a program start request is allowed to attach to a prestart job, only the program start request user ID and password and its authority to the communications device, library, and program are checked. To avoid cases where the program start request user profile is not authorized to objects to which the prestart job entry user profile is authorized, you should ensure that the user profile specified on the program start request is authorized to at least as many objects as that on the prestart job entry.

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

- Create your prestart job program when you are running under the prestart job entry user profile, and specify the value *OWNER for the user profile on the CRTXXXPGM command for your prestart jobs program, where XXX is the program language you are using (C/400,

COBOL/400, or RPG/400 programming language).

- Explicitly check for object authorization (using the Check Object (CHKOBJ) command) before you change or access any objects.

Files and objects to which a prestart job entry user profile is not authorized should be deallocated before you end your transaction.

Prestart Jobs Program

Figure 8-2 is a COBOL/400 prestart jobs program that can be used to handle program start requests. The initial processing, for example, opening the ICF file and printer file, is done first before the acquire of a requesting

program device. The acquire operation then causes the prestart job program to wait for a program start request. After acquiring the requesting program device, an accept is done to retrieve the program initialization parameters, and a CL program is called to change some of the job attributes based on the program start request user profile (using the CHGPJ command). After the main body of the program is done processing, the communications session is ended, and the program loops back to the acquire operation and waits for the next program start request to be received.

```

5738CB1 V2R1M0 910524          IBM AS/400 COBOL/400          QNETUSER/PJPGM          RCHAS400 10/03/90 15:00:41      Page 1
Program . . . . . : PJPGM
Library . . . . . : QNETUSER
Source file . . . . . : QCBLSRC
Library . . . . . : QNETUSER
Source member . . . . . : PJPGM 03/08/90 13:04:36
Generation severity level . . . . . : 29
Text 'description' . . . . . : COBOL program for prestart job example in C.P.G.
Source listing options . . . . . : *SOURCE
Generation options . . . . . : *NONE
Message limit:
  Number of messages . . . . . : *NOMAX
  Message limit severity . . . . . : 29
Print file . . . . . : QSYSVRT
Library . . . . . : *LIBL
FIPS flagging . . . . . : *NOFIPS *NOSEG *NODEB *NOOBSOLETE
SAA flagging . . . . . : *NOFLAG
Flagging severity . . . . . : 0
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Compiler . . . . . : IBM AS/400 COBOL/400
5738CB1 V2R1M0 910524          AS/400 COBOL Source          QNETUSER/PJPGM          RCHAS400 10/03/90 15:00:41      Page 2
STMT SEQNBR -A 1 B.....2.....3.....4.....5.....6.....7..IDENTFCN S COPYNAME  CHG DATE
 1 000010 IDENTIFICATION DIVISION.
 2 000030 PROGRAM-ID. PJPGM. 03/07/90
 000050*****
 000060* THIS IS A PRESTART JOB TARGET PROGRAM * 03/07/90
 000070* THIS PROGRAM WILL LOOP FOREVER * 03/07/90
 000180*****
 3 000200 ENVIRONMENT DIVISION.
 4 000220 CONFIGURATION SECTION.
 5 000240 SOURCE-COMPUTER. IBM-AS400. 03/07/90
 6 000250 OBJECT-COMPUTER. IBM-AS400. 03/07/90
 7 000260 SPECIAL-NAMES. I-O-FEEDBACK IS IO-FEEDBACK
 8 000270 OPEN-FEEDBACK IS OPEN-FBA 03/08/90
 9 000271 PIP-DATA IS PIP-PARM. 03/08/90
 000280
10 000290 INPUT-OUTPUT SECTION.
11 000310 FILE-CONTROL.
 000311*****
 000312* FILE SPECIFICATIONS * 03/07/90
 000313* ICFFILE : ICF FILE * 03/07/90
 000314* QSYSVRT : PRINTER FILE * 03/07/90
 000315*****
12 000330 SELECT ICFFILE ASSIGN TO WORKSTATION-PJDDS-SI 03/07/90
13 000340 ORGANIZATION IS TRANSACTION
14 000360 FILE STATUS IS STATUS-IND MAJ-MIN.
15 000370 SELECT QPRINT ASSIGN TO PRINTER-QSYSVRT. 03/07/90

```

Figure 8-2 (Part 1 of 3). COBOL/400 Coding for a Prestart Job Program

```

16 000420 DATA DIVISION.
17 000440 FILE SECTION.
18 000500 FD ICFFILE
19 000510 LABEL RECORDS ARE STANDARD.
20 000520 01 ICFREC.
21 000530 COPY DDS-ALL-FORMATS-I-O OF PJDDS.
22 +000001 05 PJDDS-RECORD PIC X(4101).
+000002* I-O FORMAT:INPFMT FROM FILE PJDDS OF LIBRARY QNETUSER <-ALL-FMTS
+000003* <-ALL-FMTS
23 +000004 05 INPFMT REDEFINES PJDDS-RECORD. <-ALL-FMTS
24 +000005 06 INPDATA PIC X(4096). <-ALL-FMTS
+000006* I-O FORMAT:DETACH FROM FILE PJDDS OF LIBRARY QNETUSER <-ALL-FMTS
+000007* <-ALL-FMTS
+000008* 05 DETACH REDEFINES PJDDS-RECORD. <-ALL-FMTS
+000009* I-O FORMAT:EOS FROM FILE PJDDS OF LIBRARY QNETUSER <-ALL-FMTS
+000010* <-ALL-FMTS
+000011* 05 EOS REDEFINES PJDDS-RECORD. <-ALL-FMTS
+000012* I-O FORMAT:INVITE FROM FILE PJDDS OF LIBRARY QNETUSER <-ALL-FMTS
+000013* <-ALL-FMTS
+000014* 05 INVITE REDEFINES PJDDS-RECORD. <-ALL-FMTS
+000015* I-O FORMAT:FAIL FROM FILE PJDDS OF LIBRARY QNETUSER <-ALL-FMTS
+000016* <-ALL-FMTS
+000017* 05 FAIL REDEFINES PJDDS-RECORD. <-ALL-FMTS
+000018* I-O FORMAT:IOFMT FROM FILE PJDDS OF LIBRARY QNETUSER <-ALL-FMTS
+000019* <-ALL-FMTS
25 +000020 05 IOFMT REDEFINES PJDDS-RECORD. <-ALL-FMTS
26 +000021 06 IODATA PIC X(4096). <-ALL-FMTS
27 +000022 06 OUTLEN PIC S9(5). <-ALL-FMTS
28 000590 FD QPRINT
29 000600 LABEL RECORDS ARE OMITTED.
5738CB1 V2R1M0 910524 AS/400 COBOL Source QNETUSER/PJPGM RCHAS400 10/03/90 15:00:41 Page 3
STMT SEQNBR -A 1 B. ....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
30 000610 01 PRINTREC.
31 000620 05 PRNNOTE PIC X(132).
32 000640 WORKING-STORAGE SECTION.
33 000660 77 STATUS-IND PIC X(2).
34 000661 77 PIP PIC X(6).
35 000662 01 MAJ-MIN.
36 000663 05 MAJ PIC X(2).
37 000664 05 MIN PIC X(2).
38 000840 01 10-FEEDBACK.
39 000850 05 FILLER PIC X(149).
40 000860 05 ACTUAL-LEN PIC 9(5) COMP-4.
000870*
41 001090 PROCEDURE DIVISION.
001091
001101*
001450 START-PROGRAM-PARAGRAPH.
001451*****
001452* OPEN ICF FILE AND PRINTER FILE *
001454*****
42 001470 OPEN OUTPUT QPRINT.
43 001471 OPEN I-O ICFFILE.
001480 MAIN-LOOP.
44 001481 MOVE "WAITING FOR TRANSACTION" TO PRNNOTE.
45 001482 WRITE PRINTREC.
001483*****
001484* ACQUIRING THE REQUESTER PROGRAM DEVICE CAUSES THIS PRESTART *
001485* JOB PROGRAM TO WAIT FOR A PROGRAM START REQUEST. *
001486*****
46 001590 ACQUIRE "REQDEVICE " FOR ICFFILE.
001591*****
001592* IF THIS PRESTART JOB IS BEING ENDED WITH THE CONTROLLED OPTION, *
001593* PERFORM END-OF-JOB PROCESSING AND EXIT. *
001594*****
47 001600 IF MAJ-MIN = "8209" GO TO END-JOB.
49 001610 MOVE "TRANSACTION ATTACHED" TO PRNNOTE.
50 001620 WRITE PRINTREC.
001621*****
001622* CALL A CL PROGRAM TO USE SOME JOB ATTRIBUTES FROM THE PROGRAM *
001624* START REQUEST USER PROFILE (CHGPJ). *
001625*****
51 001630 CALL "CLPGM".
001631*****
001632* MOVE THE PIP DATA FROM THE PIP DATA AREA INTO IDENTIFIER PIP. *
001634*****
52 001635 ACCEPT PIP FROM PIP-PARM END-ACCEPT.

```

Figure 8-2 (Part 2 of 3). COBOL/400 Coding for a Prestart Job Program

```

001636*****                                03/07/90
001637* MAIN BODY OF THE PROGRAM, THAT DOES THE COMMUNICATIONS I/O *      03/07/90
001638* WITH THE SOURCE PROGRAM, SHOULD BE PLACED HERE. *                03/07/90
001639*****                                03/07/90
001640*****                                03/07/90
001641* WHEN THE MAIN BODY OF THE PROGRAM IS DONE PROCESSING, END *      03/07/90
001642* THE COMMUNICATIONS SESSION AND LOOP BACK TO WAIT FOR THE NEXT *    03/07/90
001643* PROGRAM START REQUEST TO COME IN. *                                03/07/90
001644*****                                03/07/90
53 001645 WRITE ICFREC FORMAT IS "EOS".                                03/08/90
5738CB1 V2R1M0 910524 AS/400 COBOL Source QNETUSER/PJPGM RCHAS400 10/03/90 15:00:41 Page 4
STMT SEQNBR -A 1 B...2....3....4....5....6....7..IDENTFCN S COPYNAME CHG DATE
54 001650 GO TO MAIN-LOOP.                                           03/07/90
001660 END-JOB.                                                       03/07/90
55 001670 MOVE "PRESTART JOB BEING ENDED CONTROLLED" TO PRTNOTE.     03/07/90
56 001680 WRITE PRINTREC.                                             03/07/90
57 001690 CLOSE ICFFILE.                                              03/07/90
58 001691 CLOSE QPRINT.                                               03/07/90
59 001692 STOP RUN.                                                   03/07/90
001693 MAIN-EXIT.                                                     03/07/90
001700 EXIT.                                                           03/07/90
***** END OF SOURCE *****
5738CB1 V2R1M0 910524 AS/400 COBOL Messages QNETUSER/PJPGM RCHAS400 10/03/90 15:00:41 Page 5
STMT
* 21 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 000530
Message . . . . : No INPUT fields found for format DETACH.
* 21 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 000530
Message . . . . : No INPUT fields found for format EOS.
* 21 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 000530
Message . . . . : No INPUT fields found for format INVITE.
* 21 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 000530
Message . . . . : No INPUT fields found for format FAIL.
***** END OF MESSAGES *****
Message Summary
Total      Info(0-4)      Warning(5-19)      Error(20-29)      Severe(30-39)      Terminal(40-99)
4          0              4                  0                  0                  0
Source records read . . . . . : 97
Copy records read . . . . . : 22
Copy members processed . . . . : 1
Sequence errors . . . . . : 0
Highest severity message issued . . : 10
LBL0901 00 Program PJPGM created in library QNETUSER.
***** END OF COMPILATION *****

```

Figure 8-2 (Part 3 of 3). COBOL/400 Coding for a Prestart Job Program

System Considerations

The specification of the PURGE parameter on the Create Class (CRTCLS) command for the routing entry used for communications can affect communications performance. The PURGE parameter controls the way the job's operating resources are used when the job enters a wait state. If PURGE(*YES) is specified, the job's operating resources are exchanged to auxiliary storage when the job enters a wait state. If PURGE(*NO) is specified, the job's operating resources are not exchanged to auxiliary storage when the job enters a wait state.

For communications, PURGE(*YES) is normally used for interactive applications because the application is normally waiting for a response from a work station operator. The wait delay time can be long enough for another job to do useful processing with the resources. PURGE(*NO) is normally used for batch applications when the AS/400 system is sending data to

or receiving data from another computer and the wait delay will be nominal.

For a complete description of these parameters or the CRTCLS command, see the *Work Management Guide*.

Security Considerations

The security provided on the AS/400 system is used to control who can use a communications device description and its associated commands.

When a program issues an evoke to a remote AS/400 system, you must ensure that proper security information is passed on the evoke request. See Chapter 6 and Chapter 7 for a discussion on how to specify the security information on the evoke.

Although the security officer or service user has all object authority, explicit authorization to a target communications device description must be made. For example, if the security officer or

service user has not been explicitly granted authority to the target device description, the program start request is rejected by the target system. This aspect of system security is consistent with the work station security implementation for the security officer and service user profiles. Anyone with object management authority for a device description can grant authority for the device description to the security officer or service user. If the security officer or service user creates a device description, the security officer or service user (like anyone else who creates device descriptions) is explicitly authorized to the device description. When a communications device description is authorized to all users (*ALL), the security officer and service user are not included. This allows the security officer or service user to specify the device description from which the security officer or service functions can be performed.

Refer to the appropriate communications programming manual for more information about security considerations.

File Considerations

You should consider the following when deciding whether to use the system-supplied QICDMF file in your program:

- System-supplied formats can be used either with the system-supplied QICDMF file or with an ICF file made using DDS processing keywords and the CRTICFF command.
- User-defined formats cannot be added to the QICDMF file. Therefore, if your program uses the QICDMF file, it cannot use DDS processing keywords or externally described data.

- If your program uses an ICF file created using DDS and the CRTICFF command, your program can use both the record formats defined as part of your ICF file and system-supplied formats. The high-level language you are using may have some restrictions on mixing system-supplied formats and user-defined formats.

File Redirection

You can override ICF files by using the override with file commands.

When you change the file used in a program without changing the file type, the new file being used in the program is processed in the same manner as the original file. The format levels in the file must agree with the compiled program if level checking is done. If level checking is not done, the format of the changed-to file must be compatible with the compiled program or the results cannot be predicted.

If you change from one file type to another, the file-dependent characteristics of the file are ignored and certain defaults are used.

For a complete description of file redirection and the defaults used, see the *Data Management Guide*.

Additional Considerations

When using a particular communications type, you must be familiar with the requirements and restrictions unique to that communications type. For specific details, refer to the appropriate programming manual for the communications type you are using.

Chapter 9. Communications Applications with C/400 Programming Language

Previous chapters in this manual describe the functions provided by ICF. This chapter introduces you to the C/400 interface for ICF and provides program examples.

One program example is presented in this chapter, and both the source and target programs are provided. The example is a multiple-session inquiry application using one display file and four ICF sessions, and is written using user-defined formats (data description specifications or DDS).

Not all programming considerations or techniques are illustrated in the examples in this chapter. Review these examples and the examples provided in the appropriate programming manual before beginning application design and coding.

Note: The examples in this chapter were written to the APPC communications type. Minor changes might be required if another communications type is used.

Introduction to the C/400 Interface

Before you write a C/400 communications application, you must understand the high-level language interface provided by C/400 programming language.

The operations you use in the communications portion of your program are similar to work station operations. In the noncommunications portion of your program, you can use all noncommunications operations you normally use to process data that is sent or received between your program and the remote program.

Figure 9-1 briefly introduces the C/400 functions you use in the communications portion of your program.

Note: C/400 functions are case sensitive.

Refer to the *C/400* User's Guide* for details on the syntax and function of each operation.

Figure 9-1. C/400 Function

ICF Operation	C/400 Function	Function
Open	fopen, _Ropen	Opens the ICF file
Acquire	QXXACQUIRE, _Racquire	Establishes a session
Get-Attributes	QXXDEVATR, _Rdevatr	Gets the attributes of a session
Read	fread, _Rreadn	Receives data from a specific program device
Read-from-invited-program-devices	QXXREADINVDEV, followed by an fread, _Rreadindv	Receives data from any invited program device ¹
Write	fwrite, _Rwrite	Performs many of the ICF communications functions within a session
Write/Read	_Rwriterd	Performs the specified function and then receives data from the remote system.
Release	QXXRELEASE, _Rrelease	Releases the session
Close	fclose, _Rclose	Closes the ICF file

¹ The read-from-invited-program-devices operation could complete without data if the timer interval established with either the timer function or WAITRCD ends, or your job is ended (controlled).

Multiple-Session Inquiry

This example illustrates an interactive inquiry application that communicates with multiple ICF sessions. A source AS/400 system program accepts inquiries from a display device and sends a request to one of four AS/400 systems. The source program communicates with the display device through a display file, and with the four remote systems through a single ICF file.

The purpose of this example is to show multiple sessions from a single ICF file. The source program communicates with four sessions. From the viewpoint of each of the four target programs, there is only one session (with the requesting program device). Therefore, the target programs do not require any unique logic to support the multiple-session source.

Both the source program and the target program are described. The same target program is evoked in each of the four separate remote systems. Therefore, only one target program is shown in the programming example.

Error Handling

C/400 programming language provides an external variable, *Maj_Min_rc*, in the header file `<stdio.h>`, which contains the ICF major and minor codes after a read or write operation. The major and minor codes can also be obtained from the I/O feedback area.

A global variable, *ERRNO*, is defined in the header file `<errno.h>`. Two *ERRNO* values indicate that an exception has occurred:

EIOERROR

An I/O error has occurred that is not recoverable.

EIORECERR

A recoverable I/O error has occurred.

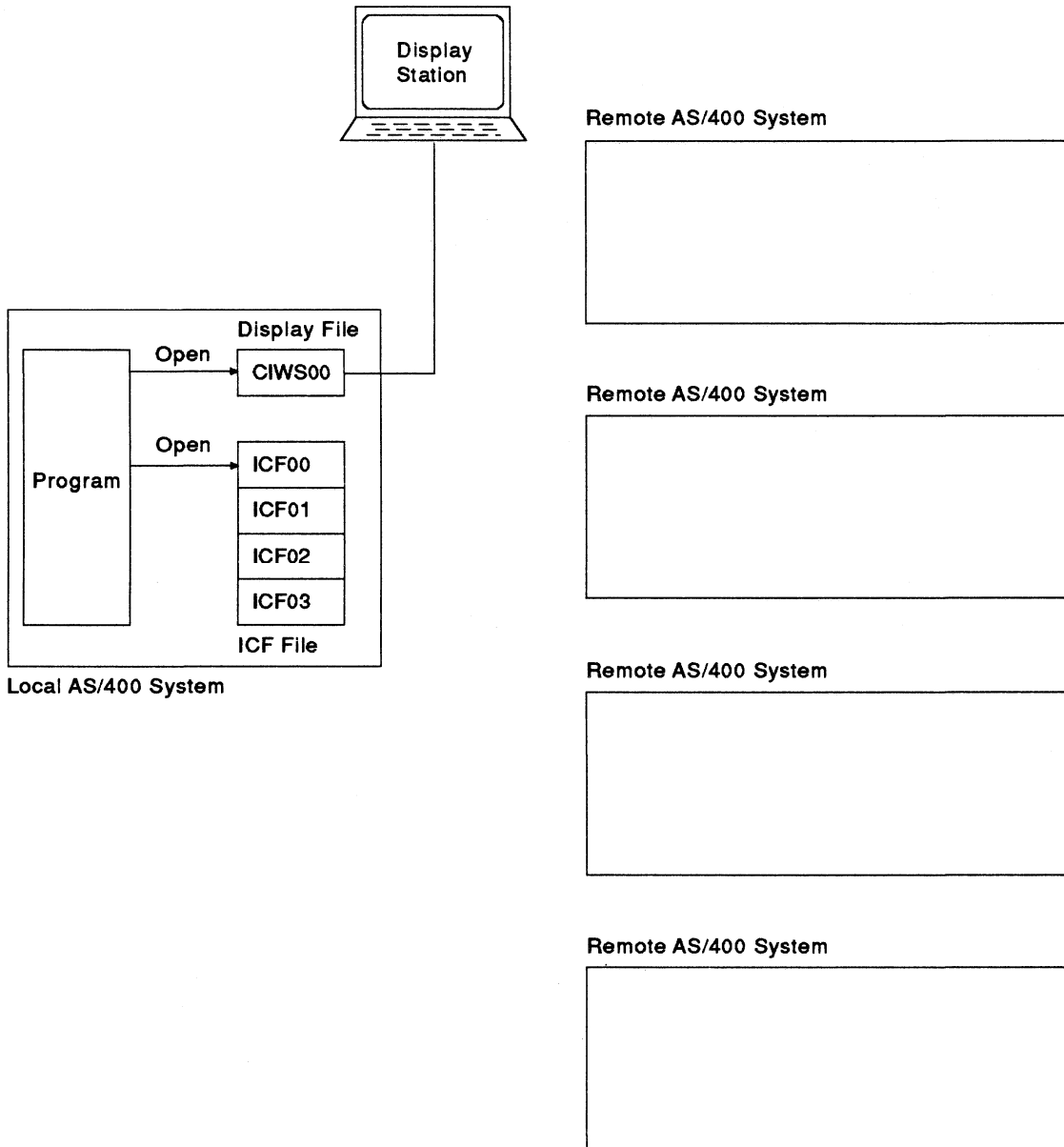
Accessing the Feedback Areas

Your program can obtain a copy of the open feedback area by using the `_Ropnfbk` routine. The `_Riofbk` routine can be used to obtain a copy of the I/O feedback area.

Transaction Flow of the Multiple-Session Inquiry:

The program shown in Figure 9-2 is started from a display station, and both the display and the ICF file are opened. CIWS00 is the *REQUESTER device, acquired when the display file opens. CIWS00 is acquired because DEV(*REQUESTER) was specified when the

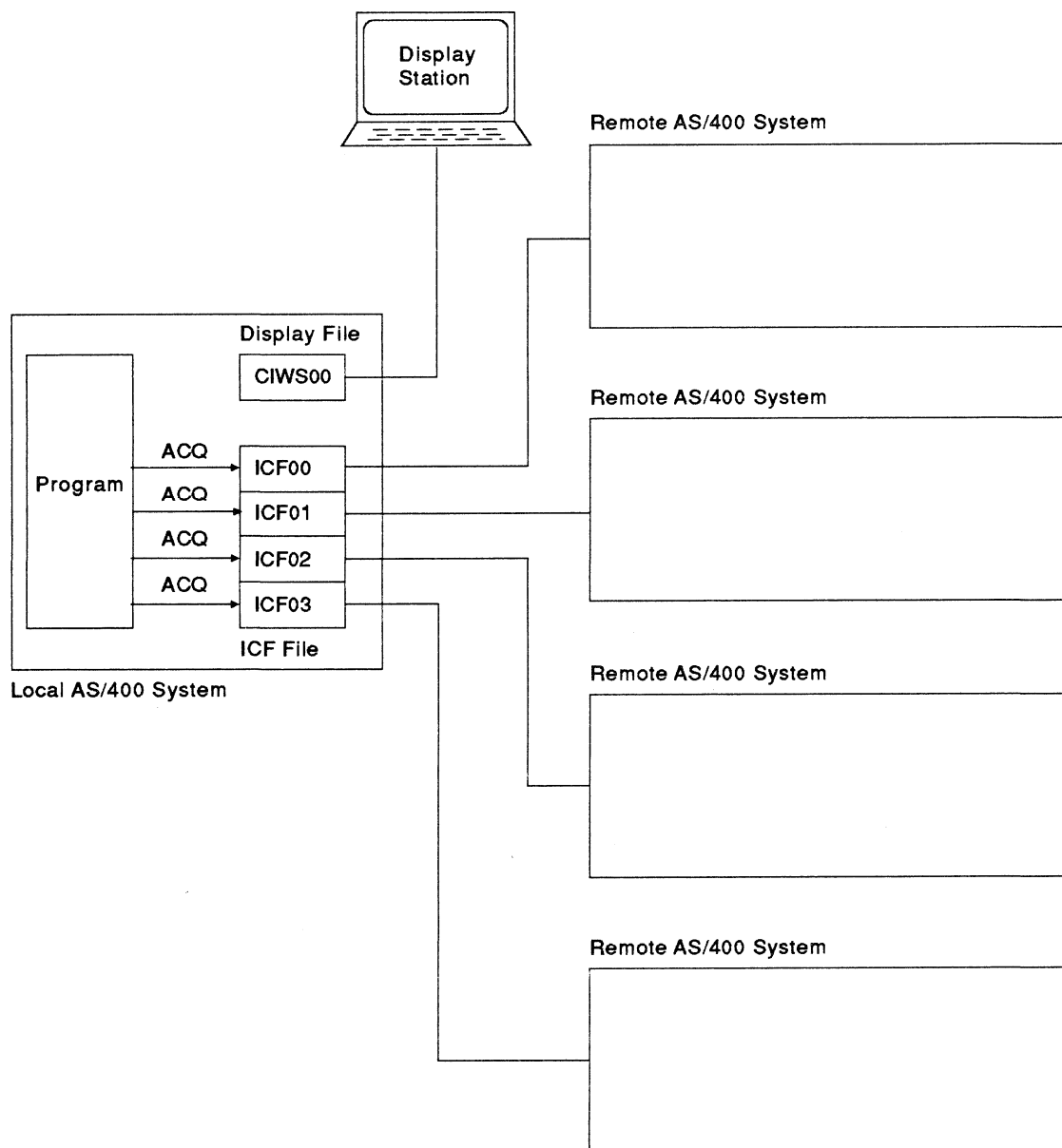
display file was created. Since the ICF file was created with ACQPGMDEV(*NONE), no ICF program devices are acquired during open processing.



RSL5100-4

Figure 9-2. Program Starts at Display Station

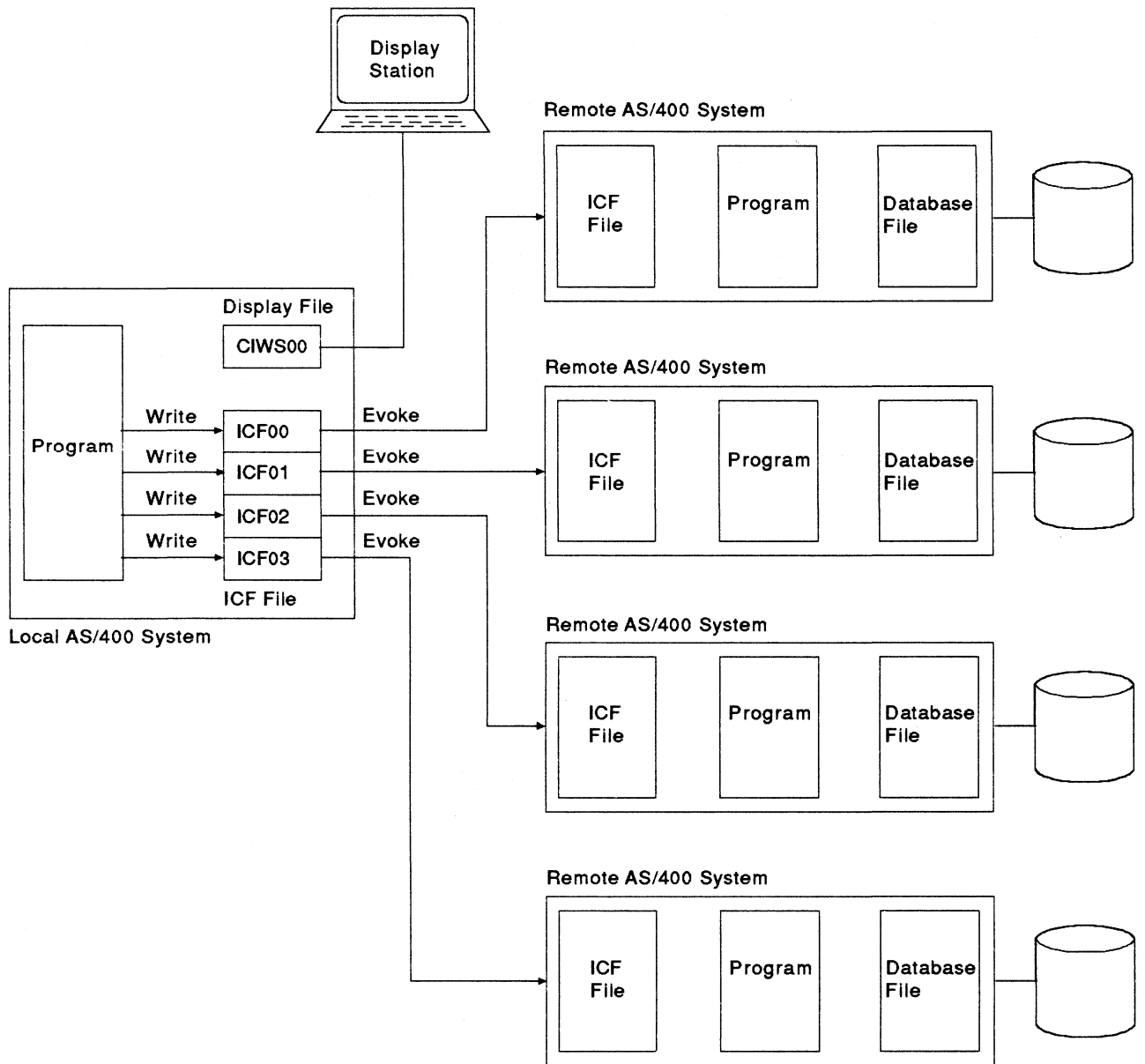
All other program devices are explicitly acquired by the program, as shown in Figure 9-3.



RSL9651-4

Figure 9-3. Program Devices Explicitly Acquired

All target programs are started with the evoke,
as shown in Figure 9-4.

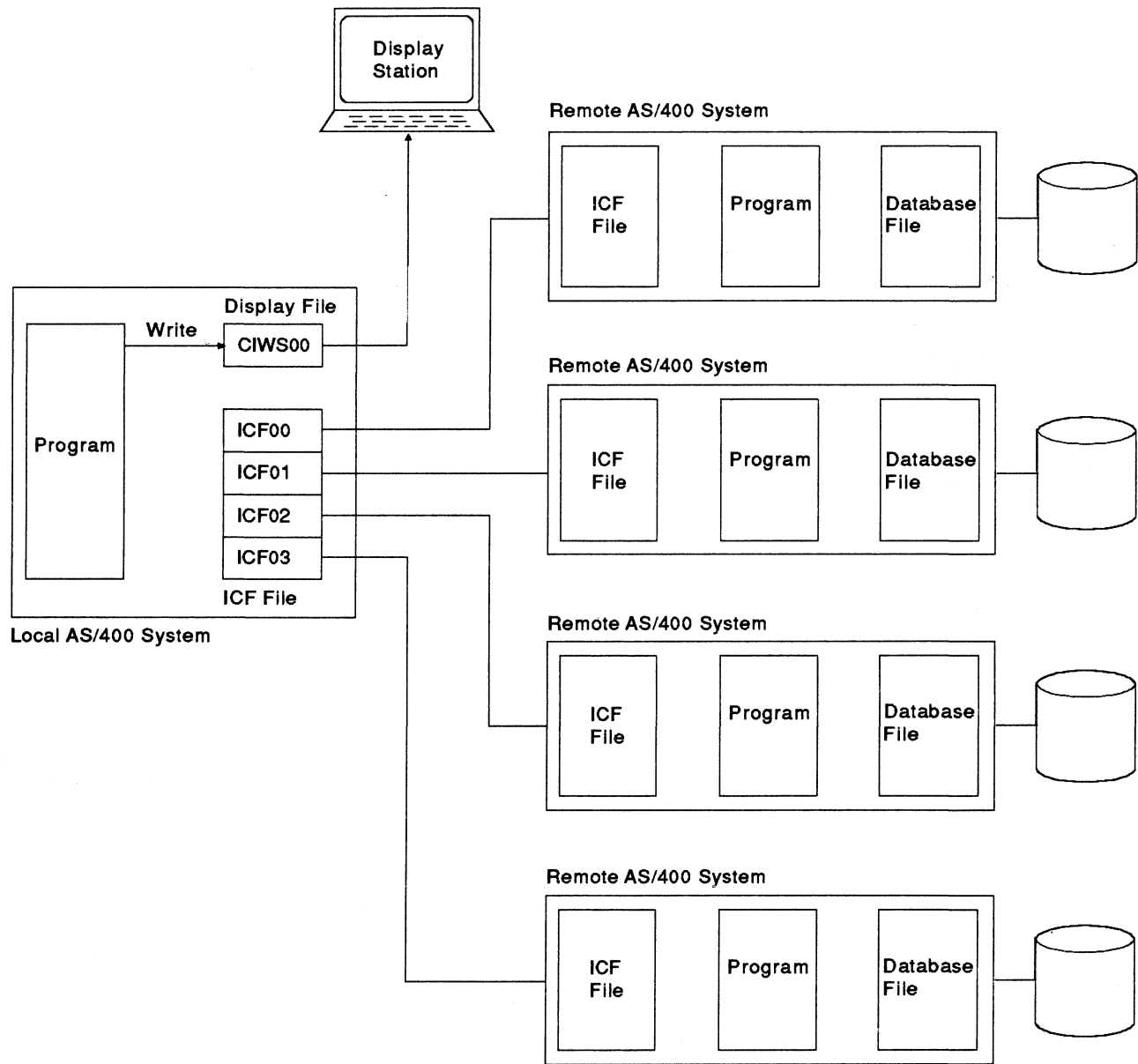


RSL6652-4

Figure 9-4. Evoke Starts Target Programs

The source program uses a specific program device name. Each target program uses an ICF file with a program device name that is associated with the requesting program device. The target program's only session is the one used to communicate with the source program. The ICF file on the remote system must be opened by the

C/400 language support using the open operation, and the requesting program device is acquired when the file is opened using the acquire operation. The main menu is written to the display station on the local system, and the program waits for a request from the display station, as shown in Figure 9-5.

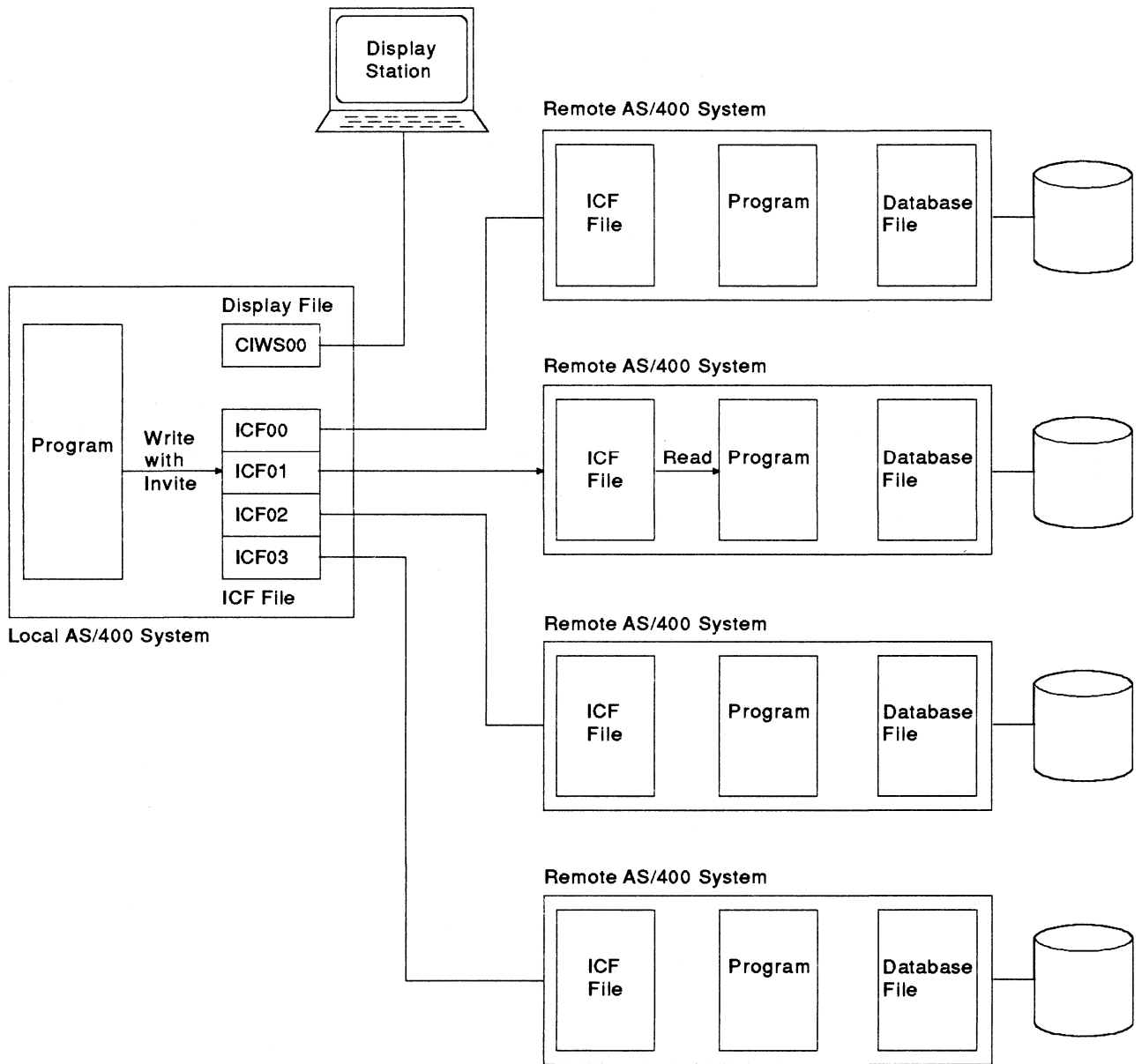


RSL9653-5

Figure 9-5. Main Menu Written to Display Station

The source program sends an inquiry request to one of the remote systems based on the request

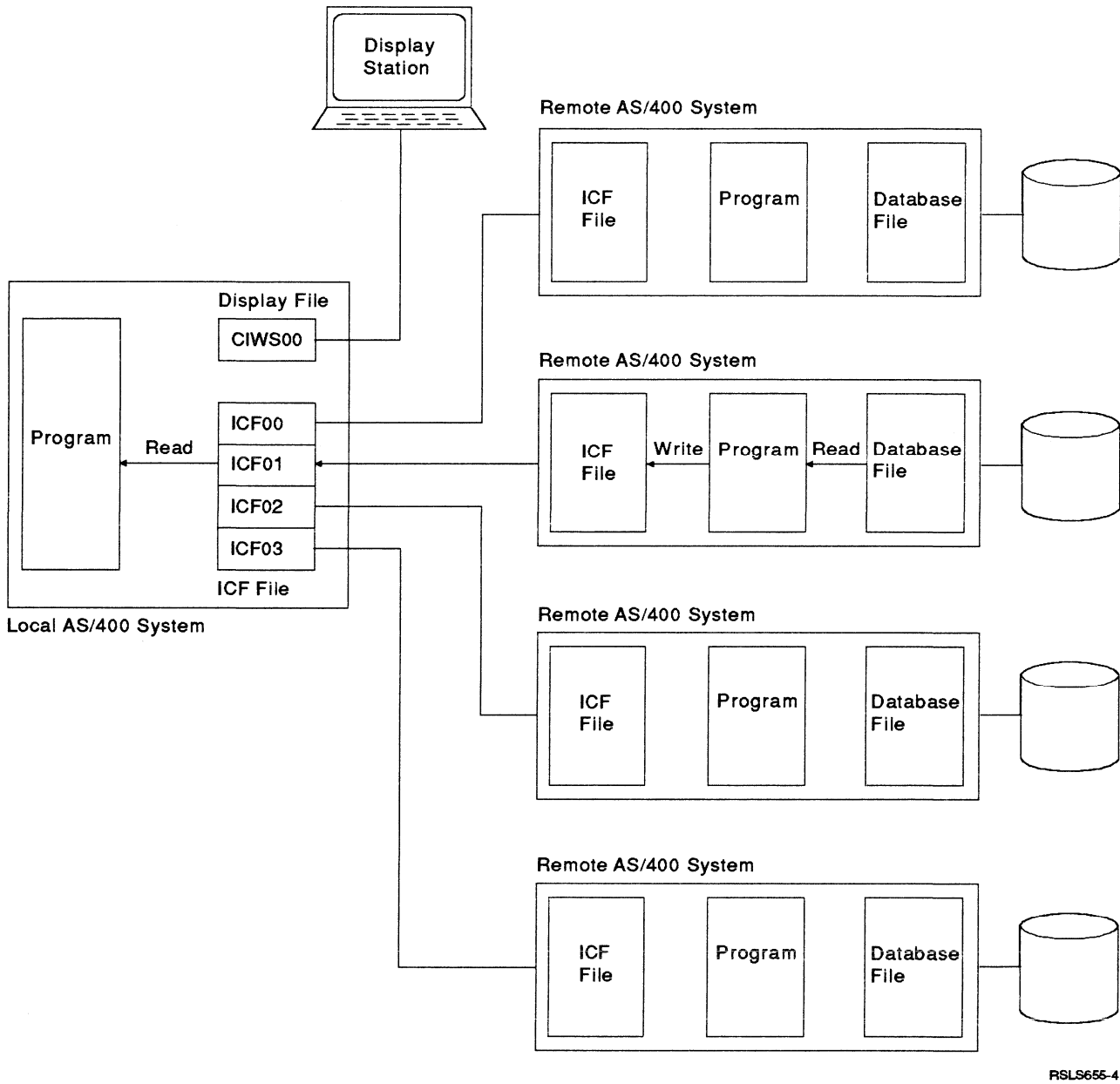
made from the display station, as shown in Figure 9-6.



RSL8654-4

Figure 9-6. Program Sends Inquiry Request to Remote System

The target program responds to the inquiry by sending a reply, as shown in Figure 9-7.

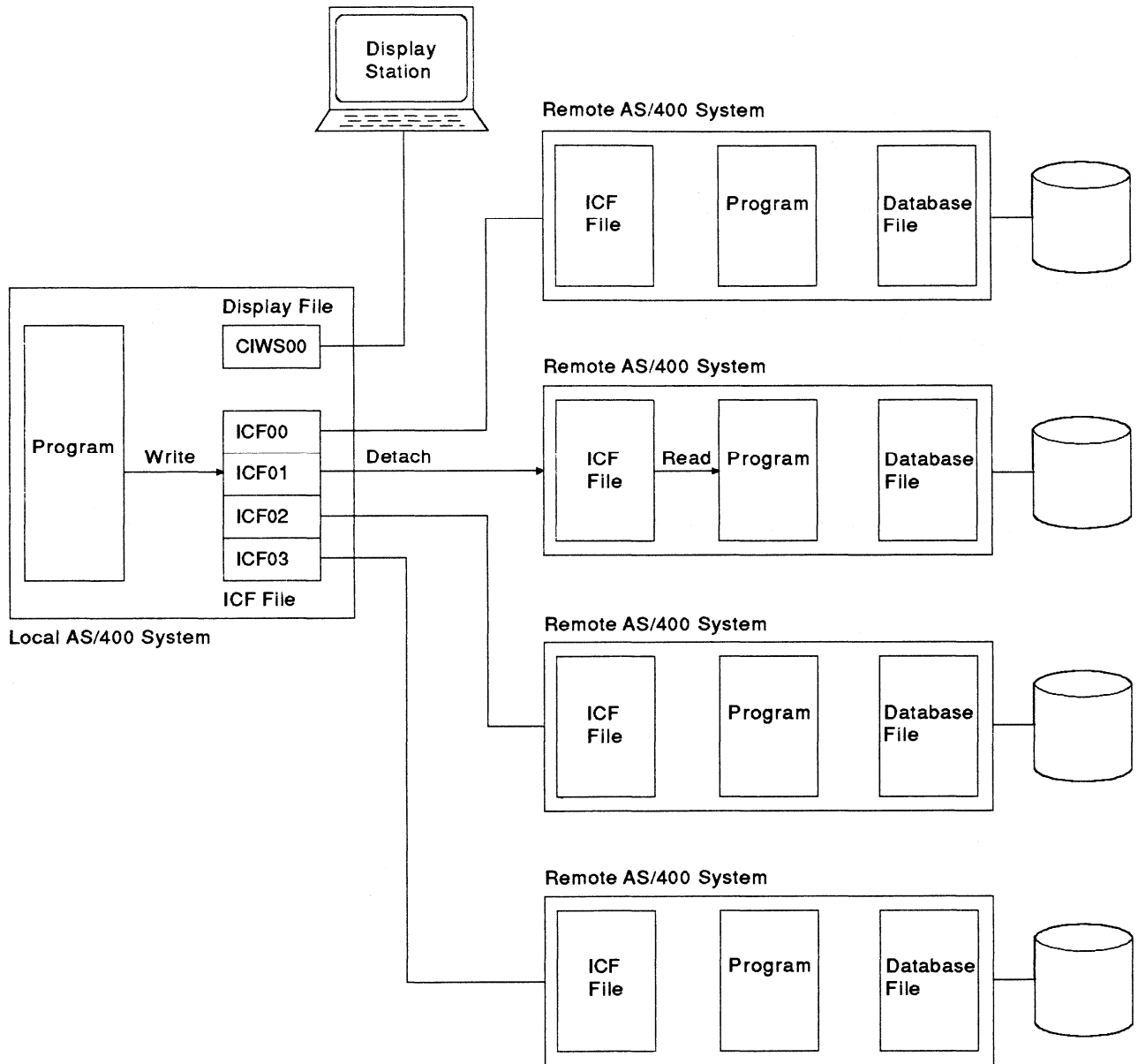


RSL655-4

Figure 9-7. Target Program Sends a Reply

The program sends a detach request and ends the session when function key 1 is pressed

(while the main inquiry menu is present), as shown in Figure 9-8.



RSL666-5

Figure 9-8. Program Ends the Session

Source Program Multiple-Session Inquiry

The following describes a C/400 source program multiple-session inquiry.

Program Files: The C/400 multiple session source program uses the following files:

CMNFIL

An ICF file used to send records to and receive records from the target program.

DSPFIL

A display file used to enter requests that are to be sent to the target program.

QSYSVRT

A printer file used to print error messages resulting from communications errors.

DDS Source: The DDS for the ICF file (CMNFIL) is illustrated below.

```

A*****
A*                                     *
A*          ICF FILE                   *
A*    USED IN SOURCE MULTIPLE SESSION PROGRAM *
A*                                     *
A*****
A          INDARA
A          R ITMRSP
A          RECITM      1
A          ITEMNO      6 0
A          DESC        30
A          QTYLST      7 0
A          QTYOH       7 0
A          QTYOD       7 0
A          QTYBO       7 0
A          UNITQ       2
A          PR01        7 2
A          PR05        7 0
A          UFRT        5 2
A          SLSTM       9 2
A          SLSTY      11 2
A          CSTTM       9 2
A          CSTTY      11 2
A          PRO         5 2
A          LOS         9 2
A          FILL1       56
A          R DTLRSP
A          RECCUS      1
A          CUSTNO      6 0
A          DNAME       30
A          DLSTOR      6 0
A          DSLSTM      9 0
A          DSPM01      9 0
A          DSPM02      9 0
A          DSPM03      9 0
A          DSTTYD     11 0
A          IDEPT       3 0
A          FILL2       57
A          R DETACH
A          DETACH
A          R EOS
A          EOS
A          R EVKREQ
A          EVOKE(CICFLIB/CTGTOMULCL)
A          SECURITY(2 *USER 3 *USER)
A          R ITMREQ
A          INVITE
A          ITEMNO      6 0
A          R DTLREQ
A          INVITE
A          CUSTNO      6 0

```

The DDS for the display file (DSPFIL) is illustrated below.

```

A*****
A*                                     *
A*          DISPLAY FILE                 *
A*    USED IN SOURCE MULTIPLE SESSION PROGRAM *
A*                                     *
A*****
A* BEGINNING MENU
A*****
A          INDARA
A          DSPSI2(*DS3)
A          CF01(99) CF02(98) CF03(97)
A          R CIMENU
A          TEXT('MENU FOR INQUIRY')
A          1 34'INQUIRY MENU'
A          3 1'Select one of the following:'
A          4 3'1. Item inquiry'
A          5 3'2. Customer inquiry'
A          11 1'Option:'
A          OPTION      1N I 11 9VALUES('1' '2')
A          19 5DFT('CMD KEY 1 - END ')
A          R DTLMNU
A          TEXT('CUSTOMER INQUIRY SCREEN 1')
A          2 2DFT('ENTER CUSTOMER')
A          CUSTNO      6N 0I 2 28
A          19 5DFT('CMD KEY 1 - END ')
A          19 23DFT(' 2 - MAIN MENU ')
A*
A*****
A* CUSTOMER INQUIRY SCREEN
A*****
A          R DTLSCR
A          TEXT('CUSTOMER INQUIRY SCR. #2')
A          1 30FT('CUST OPT LAST ORD & THIS +
A          $MTH1 $MTH2 $MTH3 THIS+
A          YTD NAME')
A          CUSTN      6N 2 2
A          DEPT       3N 0 2 9
A          DLSTR       6N 0 2 13
A          DSLSM       9N 0 2 22
A          DSPM1       9N 0 2 32
A          DSPM2       9N 0 2 42
A          DSPM3       9N 0 2 52
A          DSTYD      11N 0 2 62
A          CNAME       30 2 74
A          19 5DFT('CMD KEY 1 - END ')
A          19 23DFT(' 2 - MAIN MENU ')
A*
A*****
A* ITEM INQUIRY SCREEN
A*****
A          R ITMNU
A          TEXT('ITEM INQUIRY SCREEN ONE')
A          2 2DFT('ENTER ITEM NUMBER')
A          ITEMNO      6N 0I 2 20
A          19 5DFT('CMD KEY 1 - END ')
A          19 23DFT(' 2 - MAIN MENU ')
A*
A*****
A* ITEM DISPLAY
A*****
A          R ITMSC2
A          TEXT('ITEM INQUIRY SCREEN TWO') OVE+
A          RLAY
A          4 2DFT('DESC-')
A          DSC         30 4 8
A          5 2DFT('QUANTITY AVAILABLE')
A          QAVAIL      7N 0 5 25
A          6 11DFT('ON HAND')
A          QTYH        7N 0 6 25
A          7 11DFT('ON ORDER')
A          QTYO        7N 0 7 25
A          8 11DFT('BACK ORDER')
A          QTYB        7N 0 8 25
A          9 2DFT('UNIT OF MEASURE')
A          UNT         2 9 30
A          10 2DFT('PRICE PER UNIT')
A          PR1         7Y 2 10 24EDTCDE(3)
A          11 8DFT('QUANTITY')
A          PR5         7Y 0 11 25EDTCDE(3)
A          12 8DFT('FREIGHT')
A          UFR         5Y 2 12 26EDTCDE(3)
A          13 32DFT('MORE... ')
A          19 5DFT('CMD KEY 1 - END ')
A          19 23DFT(' 2 - MAIN MENU ')
A          19 40DFT(' 3 - ITEM MENU ')
A*
A*****
A* ITEM ADDITIONAL DISPLAY
A*****
A          R ITMSC3
A          TEXT('ITEM INQUIRY SCREEN 3 ') OVE+
A          RLAY
A          5 2DFT('SALES MONTH')
A          SLSM        9Y 2 5 16EDTCDE(1)
A          6 8DFT('Y-T-D')
A          SLSY        11Y 2 6 14EDTCDE(1)

```

```

A          7 2DFT('COSTS MONTH')
A          8 8DFT('Y-T-D')
A          9 14EDTCDE(1)
A          9 2DFT('PROFIT PCT')
A          9 22EDTCDE(1)
A          10 2DFT('LOST SALES')
A          10 14EDTCDE(1)
A          19 5DFT('CMD KEY 1 - END ')
A          19 23DFT(' 2 - MAIN MENU ')
A*****
A*  TIMEOUT SCREEN.
A*****
A          R TIMEOUT          TEXT('TIME OUT SCREEN')          OVE+
A          RLAY
A          20 2DFT('REMOTE SYSTEM TIMED OUT. ENTER+
A          1 TO TRY AGAIN OR 2 TO END.')
A          TIMRSP          1 I 20 61

```

ICF File Creation and Program Device Entry

Definition: The command needed to create the ICF file is:

```

CRITICFF FILE(CICFLIB/CMNFIL) SRCFILE(CICFLIB/QICFPUB)
SRCMBR(CMNFIL) ACQPGMDEV(*NONE) MAXPGMDEV(4)
WAITRCD(30) TEXT("SOURCE ICF FILE FOR MULTIPLE
SESSION PROGRAM")

```

The commands needed to define the four program device entries are:

```

OVRICFDEVE PGMDEV(ICF00) RMTLOCNAME(CHICAGO) FMTSLT(*RECID)
OVRICFDEVE PGMDEV(ICF01) RMTLOCNAME(NEWYORK) FMTSLT(*RECID)
OVRICFDEVE PGMDEV(ICF02) RMTLOCNAME(DETOIT) FMTSLT(*RECID)
OVRICFDEVE PGMDEV(ICF03) RMTLOCNAME(MADISON) FMTSLT(*RECID)

```

Program Explanation: The following explains the structure of the program example illustrated in Figure 9-9 on page 9-13. The ICF file used is defined by the user, and uses externally described data formats (DDS). The reference numbers in the explanation below correspond to the numbers in the following program example.

- 1** External file descriptions are included for the display file (DSPFIL).

DSPFIL is the display file used to receive user's requests and to report the information received based on the request. DSPFIL is implemented with the file-level keyword, INDARA, indicating a separate indicator area is used.

- 2** External file descriptions are included for the ICF file (CMNFIL).

CMNFIL is the ICF file used to send records to and receive records from each of the four target programs. CMNFIL is implemented with the file-level keyword, INDARA, indicating a separate indicator area is used.

- 3** This structure is used to print the major or minor return code to the print file QSYSPRT. Output to QSYSPRT is by record, so the filler field is used to blank out characters that may have been set by a previous write to the print file. See section **25**.

- 4** The variables to be used globally by the program are defined here. The common and display/ICF feedback area pointers, the file pointers, and the display file separate indicator area are defined.

- 5** The routines, except the main routine, are prototyped so the compiler knows the type of value returned and the type of parameters passed, if any.

- 6** The ICF and display files are opened for record input/output, and the printer file is opened for output. The ICF and display files are opened with the separate indicator area option specified.

- 7** The separate indicator area is initialized and defined. The variable dsp_indic is a character array of size 99.

- 8** The four program devices used by the program are explicitly acquired.

The device for the work station is implicitly acquired when the DSPFIL file is opened.

- 9** The program devices are specified before calling the evoke function four times to start transactions with the target program.

- 10** The main menu is displayed, and the user is asked to type a 1 to process a request for item information, or a 2 to process a request for customer information.

If CMD 1 is pressed, indicator 99 is set by the system, a detach is sent to the remote program, the sessions are released, the files are closed, and the program ends. The separate indicator area array must be initialized to character zeros before each input operation where indicators are checked.

- 11** This function evokes the target program. If an error occurs, the program is ended.

When the program start request is received at the remote system, CICFLIB is searched for CTGTDMULCL and that program then starts. CTGTDMULCL is a CL program that contains the following statements:

```
ADDLIBLE CICFLIB
CALL CICFLIB/CTGTDMUL
```

- 12** This procedure displays the item number inquiry display, and reads input from the user. If CMD 1 is pressed, the end-program flag is set, and the program ends on return to the main section of the program (main). If CMD 2 is pressed, control goes back to the main menu and a new main menu is displayed.

If a valid number was entered, then a function to send the item number to the remote program is called and, if no errors were encountered, a function is called to issue a read to the ICF file to get the item information.

- 13** This function selects the proper program device to use to send the item number to the target program and issues a write with an invite to the ICF file. ICF01 is used for numbers less than 400000, ICF02 for numbers greater than 400000 and less than 700000, and ICF03 for larger numbers. The number is copied from a display file structure to an ICF file structure, and then it is sent to the other program. The value returned from checking the return code (0 for 0000 return code, 1 for all other return codes) is returned to the caller.

- 14** This function gets called to receive item information from the target program after the user has specified a valid item number and that number has been sent to the target program.

The read is a read-from-invited-devices operation. The item number was previously sent with an invite.

A check is made to see if the remote system has timed out. (The wait time was specified on the CRTICFF command). A 0310 return code means a time-out has occurred after a read-from-invited-devices operation was issued. If a time-out did occur, a message is written to the display asking the user to try again (by typing 1) or to end the program (by typing 2). This function gets called again if the user types 1. If 2 is entered, control returns to **10** and the end-program flag is set.

If no data is received and the return code is 03XX, the request is sent again.

If the data returns in the wrong format, an error indication is returned to **10**.

The target program may not have found the record corresponding to the item number sent, in which case 000000 would be returned from the target as the item number. If 000000 was received, then control goes to **12** to read another number from the display.

The record received from the remote system is copied into the record used to print the information on the display, and then the information is written to the display using format ITMSC2.

If CMD 1 is pressed from this display, the end-program flag is set causing the program to end on return to the main section of the program (main). If CMD 2 is pressed, the main menu is written to the display. If CMD 3 is pressed, the item inquiry menu is written to the display. By pressing the Enter key, the profit and loss figures are calculated and written to the work station.

- 15** This procedure converts some of the values from character strings to integers, so that profit and loss figures may be calculated. The integers must then be converted back to character strings to conform with the character data types in the ICF file.

- 16** This procedure processes customer information requests, and is much like **12**. The customer inquiry display is displayed, indicators are checked, and functions are called to handle the sending and receiving of data.

- 17** This function sends the customer number to the remote system with an invite. Program device ICF00 is used for processing customer information.

- 18** This function receives information about a customer from the remote system and is structured like **14**. The major difference between this routine and the one in **14** is that if the record format is valid, a check for a 000000 customer number received from the remote system is made before any other checks. If the target program cannot find the record corresponding to the item or customer number, the record ID field is set to 'I', since the record ID does not exist.

- 19** This function calls a procedure (section **26**) to access the display/ICF feedback area. It then checks for a 00 major return code, which indicates that the previous I/O operation was successful.
- 20** This function calls a procedure (section **26**) to access the display/ICF feedback area. It then checks for a 03 major return code, which indicates that data was not received.
- 21** This function calls a procedure (section **26**) to access the display/ICF feedback area. It then checks for a 0310 return code, which indicates that the timer interval has ended.
- 22** This procedure issues a detach function to each of the program devices that were acquired, indicating to the remote systems that this program is about to end.
- 23** This procedure releases the sessions that were acquired, and then calls another procedure to close the files.
- 24** This procedure closes the ICF, display, and printer files.
- 25** This procedure retrieves the return code to be printed from the display/ICF feedback area, and then prints the appropriate message to the print file.
- 26** This procedure accesses the common I/O feedback area and the display/ICF feedback area. The pointers must be reset after each I/O operation, after which information is to be retrieved from the feedback areas. Therefore, this procedure is called before any checking of return codes is done.

```

5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 1
***** PROLOG *****
Program name . . . . . : CSRCDMUL
Library name . . . . . : CICFLIB
Source file . . . . . : QICFPUB
Library name . . . . . : CICFLIB
Source member name . . . . . : CSRCDMUL
Text Description . . . . . : Source C program for ICF Prog Guide
Compiler options . . . . . : *SOURCE *NOXREF *NOSHOWUSR *NOSHOWSYS *NOSHOWSKP *NOEXPMAC *NOAGR
                          : *NOPPONLY *NODEBUG *GEN *NOSECLVL *PRINT *LOGMSG
Language level options . . . . . : *EXTENDED
Source margins:
Left margin . . . . . : 1
Right margin . . . . . : 32767
Sequence columns:
Left Column . . . . . :
Right Column . . . . . :
Define name . . . . . :
Generation options . . . . . : *NOLIST *NOXREF *GEN *NOATR *NODUMP *NOOPTIMIZE *NOALWBND
                          : *NOANNO
Print file . . . . . : QSYSVRT
Library name . . . . . : *LIBL
Message flagging level . . . . . : 0
Compiler message:
Message limit . . . . . : *NOMAX
Message limit severity . . . . . : 30
Replace program object . . . . . : *YES
User profile . . . . . : *USER
Authority . . . . . : *CHANGE
Target Release . . . . . : *CURRENT
Last change . . . . . : 90/08/20 18:18:16
Source description . . . . . : Source C program for ICF Prog Guide
Compiler . . . . . : IBM C/400 Compiler
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 2
***** SOURCE *****
Line STMT
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9.....
1 |#pragma mapinc("dspf", "cicflib/dspf(*all)", "both", "p z") | 1
2 |#include "dspf" | 2
3 |#pragma mapinc("icff/itmrs", "cicflib/cmnfil(itmrs)", "input", "p z") | 3
4 |#pragma mapinc("icff/dtlrsp", "cicflib/cmnfil(dtlrsp)", "input", "p z") | 4
5 |#pragma mapinc("icff/itmreq", "cicflib/cmnfil(itmreq)", "output", "p z") | 5
6 |#pragma mapinc("icff/dtlreq", "cicflib/cmnfil(dtlreq)", "output", "p z") | 6
7 |#include "icff/itmrs" | 7
8 |#include "icff/dtlrsp" | 8
9 |#include "icff/itmreq" | 9
10 |#include "icff/dtlreq" | 10

```

Figure 9-9 (Part 1 of 11). Source Program Example — CSRCDMUL (User-Defined Formats)

```

11  /*-----*/
12  /* This program assigns four sessions as follows: */
13  /* 'ICF00' to inquire about a customer account before an order is */
14  /* processed. */
15  /* 'ICF01' to inquire about the inventory status of an item being */
16  /* ordered (item 000001 thru 399999). */
17  /* 'ICF02' to inquire about the inventory status of an item being */
18  /* ordered (item 400000 thru 699999). */
19  /* 'ICF03' to inquire about the inventory status of an item being */
20  /* ordered (item 700000 thru 999999). */
21  /* A display device is used to enter the request (using a customer */
22  /* and an item menu) that is sent to the remote system. */
23  /*-----*/
24
25  #define ON 1
26  #define OFF 0
27  #define ION '1' /* Indicator is set on */
28  #define IOFF '0'
29  #define ERROR 1 /* Error occurred */
30  #define NOERROR 0
31  #define NORM_END 1 /* Print normal end message */
32  #define RCD_ERR 2 /* Print wrong record error msg */
33  #define ERR_END 3 /* Print generic error message */
34  #include <stdio.h> /* Standard I/O header */
35  #include <recio.h> /* Record I/O header */
36  #include <stdlib.h> /* General utilities */
37  #include <stddef.h> /* Standard definitions */
38  #include <string.h> /* String handling utilities */
39  #include <xxfdbk.h> /* Feedback area structures */
40  #include <xxasio.h> /* Indicator area structure */
41  #include <xxcvt.h> /* EPM conversion routines */
42
43  CICFLIB_DSPFIL_CIMENU_both_t cimenu_dsp_i;
44  CICFLIB_DSPFIL_DTLMNU_both_t dtlmenu_dsp_i;
45  CICFLIB_DSPFIL_DTLSCR_both_t dtlscr_dsp_o;
46  CICFLIB_DSPFIL_ITMMNU_both_t itmnu_dsp_i;
47  CICFLIB_DSPFIL_ITMSC2_both_t itm2_dsp_o;
48  CICFLIB_DSPFIL_ITMSC3_both_t itm3_dsp_o;
49  CICFLIB_DSPFIL_TIMOUT_both_t timout_dsp_i_o;
50
51  CICFLIB_CMNFIL_ITMRSP_i_t itmrsf_icf_i;
52  CICFLIB_CMNFIL_DTLRSP_i_t dtlrsp_icf_i;
53  CICFLIB_CMNFIL_ITMREQ_o_t itmreq_icf_o;
54
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 3
Line STMT SEQNBR INCNO
54  1.....1.....2.....3.....4.....5.....6.....7.....8.....9.....
55  CICFLIB_CMNFIL_DTLREQ_o_t dtlreq_icf_o;
56  3
57  /*-----*/
58  /* Define structure used to write to the print file. */
59  /*-----*/
60  struct {
61  char major??(??);
62  char minor??(??);
63  char filler??(3??); /* Used for padding with blanks */
64  } print_rec;
65  4
66  /*-----*/
67  /* Declare global variables. */
68  /*-----*/
69
70  _RFILE *icffptr; /* Ptr to ICF file */
71  _RFILE *dspfptr; /* Ptr to display file */
72  _RFILE *prtfptr; /* Ptr to print file */
73  |XXIOFB_T *commfdbk; /* Ptr to common I/O feedback */
74  |XXIOFB_DSP_ICF_T *dsp_icf_fdbk; /* Ptr to dsp/ICF I/O feedback */
75  |char dsp_indic??(99??); /* Display separate indic area */
76  5
77  |int evoke_target(void);
78  |void process_item_req(int *);
79  |int send_item_req(void);
80  |int rec_item_info(int *);
81  |void calc_profit_loss(void);
82  |void process_cust_req(int *);
83  |int send_cust_req(void);
84  |int rec_cust_info(int *);
85  |int pos_ret_code(void);
86  |int pos_ret_code(void);
87  |int check_no_data(void);

```

Figure 9-9 (Part 2 of 11). Source Program Example – CSRCDMUL (User-Defined Formats)

88		lint check_timeout(void);		88
89		void detach(void);		89
96		void end_job(void);		90
91		void close_files(void);		91
92		void print_msg(int);		92
93		void get_access_to_fb(void);		93
94				94
95		main()		95
96		{		96
97		int end_pgm_flag = OFF; /* Signals program end request */		97
98		6 /*-----*/		98
99		/* Open ICF, display, and printer files. If an error occurs,*/		99
100		/* the program will end. */		100
101		/*-----*/		101
102		if ((icffptr = _Ropen("CICFLIB/CMNFIL", "ar+ indicators=y riofb=y"))		102
103		== NULL)		103
104		if ((dspfptr = _Ropen("CICFLIB/DSPFIL", "ar+ indicators=y riofb=y"))		104
105		== NULL)		105
106		exit(ERROR);		106
107		if ((dspfptr = _Ropen("CICFLIB/DSPFIL", "ar+ indicators=y riofb=y"))		107
5738CB1 V2R1M0 910524 IBM AS/400 C/400		CICFLIB/CSRCDMUL		RCHAS400
08/27/90 14:30:09				Page 4
Line STMT				SEQNBR INCNO
108	3	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9.....		108
109	4	== NULL) {		109
110	5	_Rclose(icffptr);		110
111	6	exit(ERROR);		111
112	7	} if ((prtftp = _Ropen("CICFLIB/QSYSPRT", "wr")) == NULL) {		112
113	8	_Rclose(icffptr);		113
114	9	_Rclose(dspfptr);		114
115	10	exit(ERROR);		115
116	11	}		116
117	12	7 /*-----*/		117
118	13	/* Set up separate indicator area for the display file. */		118
119	14	/*-----*/		119
120	15	memset(dsp_indic, IOFF, 99);		120
121	16	_Rindara(dspfptr, dsp_indic);		121
122	17	8 /*-----*/		122
123	18	/* Explicitly acquire four sessions. If an error occurs on */		123
124	19	/* any of the acquire operations then the ICF and display */		124
125	20	/* files will be closed, an error message will be printed, */		125
126	21	/* and the program will end. */		126
127	22	/*-----*/		127
128	23	_Racquire(icffptr, "ICF00");		128
129	24	if (pos_ret_code() == ERROR) {		129
130	25	close_files();		130
131	26	exit(ERROR);		131
132	27	}		132
133	28	_Racquire(icffptr, "ICF01");		133
134	29	if (pos_ret_code() == ERROR) {		134
135	30	close_files();		135
136	31	exit(ERROR);		136
137	32	}		137
138	33	_Racquire(icffptr, "ICF02");		138
139	34	if (pos_ret_code() == ERROR) {		139
140	35	close_files();		140
141	36	exit(ERROR);		141
142	37	}		142
143	38	_Racquire(icffptr, "ICF03");		143
144	39	if (pos_ret_code() == ERROR) {		144
145	40	close_files();		145
146	41	exit(ERROR);		146
147	42	}		147
148	43	_Racquire(icffptr, "ICF04");		148
149	44	if (pos_ret_code() == ERROR) {		149
150	45	close_files();		150
151	46	exit(ERROR);		151
152	47	}		152
153	48	9 /*-----*/		153
154	49	/* Evoke the target four times. If an error occurs on any of */		154
155	50	/* the evoke operations then detach, release, and close */		155
156	51	/* operations will be issued, an error message printed, and */		156
157	52	/* the program will end. */		157
158	53	/*-----*/		158
159	54	_Rpgmdev(icffptr, "ICF00");		159
160	55	if (evoke_target() == ERROR)		160
161	56	{		161
5738CB1 V2R1M0 910524 IBM AS/400 C/400		CICFLIB/CSRCDMUL		RCHAS400
08/27/90 14:30:09				Page 5
Line STMT				SEQNBR INCNO

Figure 9-9 (Part 3 of 11). Source Program Example — CSRCDMUL (User-Defined Formats)

```

162 30 | exit(ERROR);
163 31 | _Rpgmdev(icffptr, "ICF01");
164 32 | if (evoke_target() == ERROR)
165 33 | exit(ERROR);
166 34 | _Rpgmdev(icffptr, "ICF02");
167 35 | if (evoke_target() == ERROR)
168 36 | exit(ERROR);
169 37 | _Rpgmdev(icffptr, "ICF03");
170 38 | if (evoke_target() == ERROR)
171 39 | exit(ERROR);
172 |
173 | 10
174 | /*-----*/
175 | /* Put out the main menu to the display, and depending on */
176 | /* the input, either request item information or customer */
177 | /* information from remote system. If CMD 1 (indicator 99) */
178 | /* is pressed on any screen, the program ends. If the user */
179 | /* picks option 1, an item inquiry is processed, and if the */
180 | /* user picks option 2 a customer inquiry is processed. If */
181 | /* CMD 1 wasn't pressed, nor option 1 or 2, then the input */
182 | /* is invalid. */
183 | /*-----*/
184 40 | while (end_pgm_flag == OFF) {
185 41 |     _Rformat(dspfp, "CIMENU");
186 42 |     _Rwrite(dspfp, NULL, 0);
187 43 |     memset(dsp_indic, IOFF, 99);
188 44 |     _Rreadn(dspfp, &cimenu_dsp_i, sizeof(cimenu_dsp_i), __DFT);
189 45 |     if (dsp_indic??(98??) == IOFF) {
190 46 |         if (cimenu_dsp_i.OPTION == '1')
191 47 |             process_item_req(&end_pgm_flag);
192 |         else
193 48 |             if (cimenu_dsp_i.OPTION == '2')
194 49 |                 process_cust_req(&end_pgm_flag);
195 |         }
196 |         else {
197 50 |             print_msg(NORM_END);
198 51 |             end_pgm_flag = ON;
199 |         }
200 |     }
201 52 |     detach();
202 53 |     end_job();
203 | }
204 |
205 |
206 | /*-----*/
207 | /* Evoked the Target Program */
208 | /* Evoked the target program. If an error occurs then a detach will be */
209 | /* sent, a release will be issued, and the ICF, display, and printer */
210 | /* files will be closed. */
211 | /*-----*/
212 | 11
213 | evoke_target()
214 | {
215 |     _Rformat(icffptr, "EVKREQ");
5738CB1 V2RIM0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 6
Line STMT SEQNBR INCNO
216 2 | _Rwrite(icffptr, NULL, 0);
217 3 | if (pos_ret_code() == ERROR) {
218 4 |     print_msg(ERR_END);
219 5 |     detach();
220 6 |     end_job();
221 7 |     return(ERROR);
222 | }
223 8 | return(NOERROR);
224 | }
225 |
226 |
227 | /*-----*/
228 | /* Process Item Request Screen */
229 | /* This routine puts out the item request screen and reads the input */
230 | /* from the user. CMD 1 (end job), CMD 2 (go to main menu), and CMD 3 */
231 | /* (reenter item number) are checked. Routines to send a valid item */
232 | /* number and to receive item information are called. */
233 | /*-----*/
234 | 12
235 | void process_item_req(int *end_pgm_flag)
236 | {
237 |     _Rformat(dspfp, "ITMMNU");
238 |     _Rwrite(dspfp, NULL, 0);

```

Figure 9-9 (Part 4 of 11). Source Program Example — CSRCDMUL (User-Defined Formats)

```

239 3 | memset(dsp_indic, IOFF, 99);
240 4 | _Readn(dspfptr, &itmmnu_dsp_i, sizeof(itmmnu_dsp_i), __DFT);
241 5 | if (dsp_indic??(98??) == ION) { /* CMD 1, indic 99 */
242 6 |     print_msg(NORM_END);
243 7 |     *end_pgm_flag = ON;
244 | }
245 | else
246 8 |     if (dsp_indic??(97??) == IOFF) /* CMD 2, indic 98 */
247 9 |         if (send_item_req() == NOERROR) {
248 10 |             if (rec_item_info(end_pgm_flag) == ERROR) {
249 11 |                 print_msg(ERR_END);
250 12 |                 *end_pgm_flag = ON;
251 |             }
252 |         }
253 |     else {
254 13 |         print_msg(ERR_END);
255 14 |         *end_pgm_flag = ON;
256 |     }
257 | }
258 |
259 |
260 | /*-----*/
261 | /*          Send Item Request          */
262 | /* This routine sends the item number entered to the appropriate target */
263 | /* program based on the range of the number. */
264 | /*-----*/
265 | 13
266 | send_item_req()
267 | {
268 1 |     _Rformat(icffptr, "ITMREQ");
269 2 |     if (strncmp(itmmnu_dsp_i.ITEMNO, "399999", 6) != 1)
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 7
Line STMT SEQNBR INCNO
270 3 | *...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9..... 270
271 |     _Rpgmdev(icffptr, "ICF01");
272 |     else
273 4 |         if (strncmp(itmmnu_dsp_i.ITEMNO, "699999", 6) != 1)
274 5 |             _Rpgmdev(icffptr, "ICF02");
275 |         else
276 6 |             _Rpgmdev(icffptr, "ICF03");
277 7 |         strncpy(itmreq_icf_o.ITEMNO, itmmnu_dsp_i.ITEMNO, 6);
278 8 |         _Rwrite(icffptr, &itmreq_icf_o, sizeof(itmreq_icf_o));
279 9 |         return(pos_ret_code());
280 |     }
281 | }
282 |
283 | /*-----*/
284 | /*          Receive Item Information          */
285 | /* The item information is read from the remote system. A check is */
286 | /* made for three conditions following the read: 1) The remote system */
287 | /* timed out, 2) no data was received, and 3) data was returned in an */
288 | /* unexpected format. */
289 | /* If the remote system times out (maj/min 0310), a message is written */
290 | /* to the screen asking to try again (enter 1) or to end the program */
291 | /* (enter 2). */
292 | /* If no data is received (major 03) the request is sent again to the */
293 | /* remote system. */
294 | /* If the record returns with the wrong record format, the program will */
295 | /* end on error. */
296 | /* If CMD 1 (in99) is pressed, the end program flag is set causing the */
297 | /* program to end upon return to main. If CMD 2 (in98) is pressed, */
298 | /* the main menu is written to the screen. If CMD 3 (in97) is pressed, */
299 | /* the item inquiry menu is written to the screen. */
300 | /* If the remote program didn't find the item, and the item number */
301 | /* "000000" was returned, then the item request screen is displayed. */
302 | /*-----*/
303 | 14
304 | rec_item_info(int *end_pgm_flag)
305 | {
306 1 |     _Rreadindv(icffptr, &itmrspl_icf_i, sizeof(itmrspl_icf_i), __DFT);
307 3 |     if (check_timeout() == 0) {
308 4 |         _Rformat(dspfptr, "TIMOUT");
309 5 |         _Rwrite(dspfptr, NULL, 0);
310 6 |         _Readn(dspfptr, &timeout_dsp_i_o, sizeof(timeout_dsp_i_o), __DFT);
311 7 |         if (timeout_dsp_i_o.TIMRSP == '1')
312 8 |             return(rec_item_info(end_pgm_flag));
313 |         else
314 9 |             if (timeout_dsp_i_o.TIMRSP == '2') {
315 10 |                 print_msg(NORM_END);
316 11 |                 *end_pgm_flag = ON;

```

Figure 9-9 (Part 5 of 11). Source Program Example – CSRCDMUL (User-Defined Formats)

```

316 12 |         return(NOERROR);
317 |     }
318 | }
319 | else
320 13 |     if (check_no_data() == 0) { /* No data received */
321 14 |         _Rformat(icffptr, "ITMREQ");
322 15 |         _Rwrite(icffptr, &itmreq_icf_o, sizeof(itmreq_icf_o));
323 16 |         return(pos_ret_code());
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 8
Line STMT SEQNBR INCNO
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9.....
324 |     }
325 |     else { /* Check for valid record fmt */
326 17 |         comm_fdbk = Riofbk(icffptr);
327 18 |         if (strncmp(comm_fdbk->rec_format, "ITMRSP ", 10) != 0) {
328 19 |             print_msg(RCD_ERR);
329 20 |             return(ERROR);
330 |         }
331 |     }
332 21 |     if (strncmp(itmrsp_icf_i.ITEMNO, "000000", 6) != 1) /* Item not found */
333 22 |         process_item_req(end_pgm_flag);
334 |     else {
335 23 |         strncpy(itmsc2_dsp_o.DSC, itmrsp_icf_i.DESC, 39);
336 24 |         strncpy(itmsc2_dsp_o.QAVAIL, itmrsp_icf_i.QTYLST, 7);
337 25 |         strncpy(itmsc2_dsp_o.QTYO, itmrsp_icf_i.QTYO0, 7);
338 26 |         strncpy(itmsc2_dsp_o.QTYH, itmrsp_icf_i.QTYOH, 7);
339 27 |         strncpy(itmsc2_dsp_o.QTYB, itmrsp_icf_i.QTYBO, 7);
340 28 |         strncpy(itmsc2_dsp_o.UNT, itmrsp_icf_i.UNITQ, 2);
341 29 |         strncpy(itmsc2_dsp_o.PR1, itmrsp_icf_i.PR01, 7);
342 30 |         strncpy(itmsc2_dsp_o.PR5, itmrsp_icf_i.PR05, 7);
343 31 |         strncpy(itmsc2_dsp_o.UFR, itmrsp_icf_i.UFRT, 5);
344 32 |         _Rformat(dspfptr, "ITMSC2"); /* Display item screen 2 */
345 33 |         _Rwrite(dspfptr, &itmsc2_dsp_o, sizeof(itmsc2_dsp_o));
346 34 |         memset(dsp_indic, IOFF, 99);
347 35 |         _Rreadn(dspfptr, NULL, 0, __DFT);
348 36 |         if (dsp_indic??(98??) == ION) { /* CMD 1, indic 99 */
349 37 |             print_msg(NORM_END);
350 38 |             *end_pgm_flag = ON;
351 |         }
352 |     else
353 39 |         if (dsp_indic??(96??) == ION) /* CMD 3, indic 97 */
354 40 |             process_item_req(end_pgm_flag);
355 |     else
356 41 |         if (dsp_indic??(97??) == IOFF) {
357 42 |             calc_profit_loss();
358 43 |             _Rformat(dspfptr, "ITMSC3"); /* Display screen 3 */
359 44 |             _Rwrite(dspfptr, &itmsc3_dsp_o, sizeof(itmsc3_dsp_o));
360 45 |             memset(dsp_indic, IOFF, 99);
361 46 |             _Rreadn(dspfptr, NULL, 0, __DFT);
362 |         }
363 |     }
364 47 |     return(NOERROR);
365 | }
366 |
367 |
368 | /*-----*/
369 | /* Calculate Profit/Loss */
370 | /* This routine calculates profit and loss figures and displays them */
371 | /* on screen two of the item. */
372 | /*-----*/
373 | 15
374 | void calc_profit_loss()
375 | {
376 |     long qtylst_l;
377 |     double slstm_d, csttm_d;
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 9
Line STMT SEQNBR INCNO
*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9.....
378 |     double pr01_d, losts_d, profit_d;
379 |
380 1 |     csttm_d = QXXZT0D(itmrsp_icf_i.CSTTM, 9, 2);
381 2 |     qtylst_l = QXXZT0I(itmrsp_icf_i.QTYLST, 7, 0);
382 3 |     pr01_d = QXXZT0D(itmrsp_icf_i.PR01, 7, 2);
383 4 |     slstm_d = QXXZT0D(itmrsp_icf_i.SLSTM, 9, 2);
384 5 |     profit_d = csttm_d - slstm_d;
385 6 |     profit_d *= 100;
386 7 |     if (slstm_d > 0)
387 8 |         profit_d /= slstm_d;
388 9 |     losts_d = qtylst_l * pr01_d;
389 10 |     strncpy(itmsc3_dsp_o.SLSM, itmrsp_icf_i.SLSTM, 9);

```

Figure 9-9 (Part 6 of 11). Source Program Example — CSRCDMUL (User-Defined Formats)

```

390 11 | strncpy(itmsc3_dsp_o.SLSY, itmrsp_icf_i.SLSTY, 11);          390
391 12 | strncpy(itmsc3_dsp_o.CSTM, itmrsp_icf_i.CSTTM, 9);         391
392 13 | strncpy(itmsc3_dsp_o.CSTY, itmrsp_icf_i.CSTTY, 11);       392
393 14 | QXXD0Z(itmsc3_dsp_o.LOSTS, 11, 2, losts_d);               393
394 15 | QXXD0Z(itmsc3_dsp_o.PROFIT, 5, 2, profit_d);              394
395 | }                                                            395
396 |                                                              396
397 |                                                              397
398 | /*-----*/                                               398
399 | /*          Process Customer Request                        */  399
400 | /* This routine puts out the customer request screen and reads the */  400
401 | /* input from the user.  CMD 1 (end job) and CMD 2 (go to main menu) */  401
402 | /* are checked.  Routines to send the customer number and to receive */  402
403 | /* customer information are called.                          */  403
404 | /*-----*/                                               404
405 | 16                                                         405
406 | void process_cust_req(int *end_pgm_flag)                    406
407 | {                                                            407
408 |     1 | _Rformat(dspfptr, "DTLMNU");                             408
409 |     2 | _Rwrite(dspfptr, NULL, 0);                             409
410 |     3 | memset(dsp_indic, IOFF, 99);                           410
411 |     4 | _Rreadn(dspfptr, &dtlmu_dsp_i, sizeof(dtlmu_dsp_i), __DFT); 411
412 |     5 | if (dsp_indic??(98??) == ION) { /* CMD 1, indic 99 */ 412
413 |         6 | print_msg(NORM_END);                             413
414 |         7 | *end_pgm_flag = ON;                               414
415 |     }                                                         415
416 |     else                                                       416
417 |         8 | if (dsp_indic??(97??) == IOFF) /* CMD 2, indic 98 */ 417
418 |             9 | if (send_cust_req() == NOERROR) {           418
419 |                 10 | if (rec_cust_info(end_pgm_flag) == ERROR) { 419
420 |                     11 | print_msg(ERR_END);                 420
421 |                     12 | *end_pgm_flag = ON;                 421
422 |                 }                                             422
423 |             }                                                 423
424 |             else {                                           424
425 |                 13 | print_msg(ERR_END);                     425
426 |                 14 | *end_pgm_flag = ON;                     426
427 |             }                                                 427
428 |     }                                                         428
429 | }                                                            429
430 |                                                              430
431 | /*-----*/                                               431
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 10
Line STMT *...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9..... SEQNBR INCNO
432 | /*          Send Customer Request                          */  432
433 | /* This routine sends the customer number entered by the user to the */  433
434 | /* remote program.  The number has already been read in.          */  434
435 | /*-----*/                                               435
436 | 17                                                         436
437 | send_cust_req()                                           437
438 | {                                                            438
439 |     1 | strncpy(dtlreq_icf_o.CUSTNO, dtlmu_dsp_i.CUSTNO, 6); 439
440 |     2 | _Rformat(icffptr, "DTLREQ");                         440
441 |     3 | _Rpgmdev(icffptr, "ICF00");                         441
442 |     4 | _Rwrite(icffptr, &dtlreq_icf_o, sizeof(dtlreq_icf_o)); 442
443 |     5 | return(pos_ret_code());                             443
444 | }                                                            444
445 |                                                              445
446 |                                                              446
447 | /*-----*/                                               447
448 | /*          Receive Customer Information                    */  448
449 | /* This routine attempts to receive customer information from the */  449
450 | /* target program.  A check is made for the following three conditions */  450
451 | /* following the read operation: 1) The remote system timed out, 2) no */  451
452 | /* data was received, and 3) data returned in an unexpected format. */  452
453 | /* If the remote system times out (maj/min 0310), a message is written */  453
454 | /* to the screen asking user to try again (enter 1) or to end the */  454
455 | /* program (enter 2).                                          */  455
456 | /* If no data is received (major 03) the request is sent again to the */  456
457 | /* remote system.                                             */  457
458 | /* If the record returns with the wrong record format, the program will */  458
459 | /* end on error.                                             */  459
460 | /* If the remote program didn't find the customer, and the item number */  460
461 | /* "000000" was returned, the main menu is displayed.        */  461
462 | /*-----*/                                               462
463 | 18                                                         463
464 | rec_cust_info(int *end_pgm_flag)                            464
465 | {                                                            465
466 |     1 | _Rreadindv(icffptr, &dtlrsp_icf_i, sizeof(dtlrsp_icf_i), __DFT); 466

```

Figure 9-9 (Part 7 of 11). Source Program Example — CSRCDMUL (User-Defined Formats)

```

467 3 | if (check_timeout() == 0) {
468 4 |     _Rformat(dspfptr, "TIMEOUT");
469 5 |     _Rwrite(dspfptr, NULL, 0);
470 6 |     _Rreadn(dspfptr, &timeout_dsp_i_o, sizeof(timeout_dsp_i_o), __DFT);
471 7 |     if (timeout_dsp_i_o.TIMRSP == '1')
472 8 |         return(rec_cust_info(end_pgm_flag));
473 |     else
474 9 |         if (timeout_dsp_i_o.TIMRSP == '2') {
475 10 |             print_msg(NORM_END);
476 11 |             *end_pgm_flag = ON;
477 12 |             return(NOERROR);
478 |         }
479 |     }
480 | else
481 13 |     if (check_no_data() == 0) { /* No data received */
482 14 |         _Rformat(icffptr, "DTLREQ");
483 15 |         _Rwrite(icffptr, &dtlreq_icf_o, sizeof(dtlreq_icf_o));
484 16 |         return(pos_ret_code());
485 |     }
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 11
Line STMT SEQNBR INCNO
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9.....
486 | else { /* Was item found ? */
487 17 |     if (strncmp(dtlrsp_icf_i.CUSTNO, "000000", 6) != 1)
488 18 |         return(NOERROR);
489 |     else { /* Check record format */
490 19 |         comm_fdbk = _Riofbk(icffptr);
491 20 |         if (strncmp(comm_fdbk->rec_format, "DTLRSP ", 10) != 0) {
492 21 |             print_msg(RCD_ERR);
493 22 |             return(ERROR);
494 |         }
495 |     }
496 | }
497 23 | strncpy(dtlscr_dsp_o.CUSTN, dtlrsp_icf_i.CUSTNO, 6);
498 24 | strncpy(dtlscr_dsp_o.CNAME, dtlrsp_icf_i.DNAME, 30);
499 25 | strncpy(dtlscr_dsp_o.DLSTR, dtlrsp_icf_i.DLSTOR, 6);
500 26 | strncpy(dtlscr_dsp_o.DSLSM, dtlrsp_icf_i.DSLSTM, 9);
501 27 | strncpy(dtlscr_dsp_o.DSPM1, dtlrsp_icf_i.DSPM01, 9);
502 28 | strncpy(dtlscr_dsp_o.DSPM2, dtlrsp_icf_i.DSPM02, 9);
503 29 | strncpy(dtlscr_dsp_o.DSTYD, dtlrsp_icf_i.DSTTYD, 11);
504 30 | strncpy(dtlscr_dsp_o.DEPT, dtlrsp_icf_i.IDEPT, 3);
505 31 | _Rformat(dspfptr, "DTLSCR"); /* Display customer info */
506 32 | _Rwrite(dspfptr, &dtlscr_dsp_o, sizeof(dtlscr_dsp_o));
507 33 | memset(dsp_indic, IOFF, 99);
508 34 | _Rreadn(dspfptr, NULL, 0, __DFT);
509 35 | if (dsp_indic??(98??) == ION) {
510 36 |     print_msg(NORM_END);
511 37 |     *end_pgm_flag = ON;
512 | }
513 38 | return(NOERROR);
514 | }
515 |
516 |
517 | /*-----*/
518 | /* Check For Successful Operation */
519 | /* If the major return code is 00 the last operation attempted was */
520 | /* successful, otherwise an error occurred. */
521 | /*-----*/
522 | 19
523 | pos_ret_code()
524 | {
525 | 1 | get_access_to_fb();
526 | 2 | if (strncmp(dsp_icf_fdbk->major_ret_code, "00", 2) == 0)
527 | 3 |     return(NOERROR);
528 |     else
529 | 4 |     return(ERROR);
530 | }
531 |
532 |
533 | /*-----*/
534 | /* Check for No Data Received */
535 | /* If the major return code is 03 then no data was received on an input */
536 | /* operation. */
537 | /*-----*/

```

Figure 9-9 (Part 8 of 11). Source Program Example — CSRCDMUL (User-Defined Formats)

538		20								538
539		check_no_data()								539
5738CB1	V2R1M0	910524	IBM AS/400 C/400	CICFLIB/CSRCDMUL	RCHAS400	08/27/90	14:30:09	Page	12	
Line	STMT							SEQNBR	INCNO	
		*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9.....								
540		{								540
541	1		get_access_to_fb();							541
542	2		if (strncmp(dsp_icf_fdbk->major_ret_code, "03", 2) == 0)							542
543	3		return(0);							543
544			else							544
545	4		return(1);							545
546			}							546
547										547
548										548
549			/*-----*/							549
550			/* Check for Timeout */							550
551			/* If the major/minor return code is 0310, then a timeout occurred when */							551
552			/* reading from an invited program device. Return 1 if timeout occurred*/							552
553			/* otherwise return 0. */							553
554			/*-----*/							554
555			21							555
556			check_timeout()							556
557			{							557
558	1		get_access_to_fb();							558
559			if ((strncmp(dsp_icf_fdbk->major_ret_code, "03", 2) == 0) &&							559
560	2		(strncmp(dsp_icf_fdbk->minor_ret_code, "10", 2) == 0))							560
561	3		return(0);							561
562			else							562
563	4		return(1);							563
564			}							564
565										565
566										566
567			/*-----*/							567
568			/* Issue Detach */							568
569			/* A detach is sent to each program device to end the transaction with */							569
570			/* the remote system. */							570
571			/*-----*/							571
572			22							572
573			void detach()							573
574			{							574
575	1		_Rformat(icffptr, "DETACH");							575
576	2		_Rpgmdev(icffptr, "ICF00");							576
577	3		_Rwrite(icffptr, NULL, 0);							577
578	4		_Rpgmdev(icffptr, "ICF01");							578
579	5		_Rwrite(icffptr, NULL, 0);							579
580	6		_Rpgmdev(icffptr, "ICF02");							580
581	7		_Rwrite(icffptr, NULL, 0);							581
582	8		_Rpgmdev(icffptr, "ICF03");							582
583	9		_Rwrite(icffptr, NULL, 0);							583
584			}							584
585										585
586										586
587			/*-----*/							587
588			/* End the Job */							588
589			/* A release is sent to each program device to end the sessions, and */							589
590			/* the ICF, display, and printer files are closed. */							590
591			/*-----*/							591
592			23							592
593			void end_job()							593
5738CB1	V2R1M0	910524	IBM AS/400 C/400	CICFLIB/CSRCDMUL	RCHAS400	08/27/90	14:30:09	Page	13	
Line	STMT							SEQNBR	INCNO	
		*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9.....								
594		{								594
595	1		_Rrelease(icffptr, "ICF00");							595
596	2		_Rrelease(icffptr, "ICF01");							596
597	3		_Rrelease(icffptr, "ICF02");							597
598	4		_Rrelease(icffptr, "ICF03");							598
599	5		close_files();							599
600			}							600
601										601
602										602
603			/*-----*/							603
604			/* Close the Files */							604
605			/* Close the ICF, display, and print files. */							605
606			/*-----*/							606
607			24							607
608			void close_files()							608
609			{							609
610	1		_Rclose(icffptr);							610
611	2		_Rclose(dspfpfptr);							611
612	3		_Rclose(prtfpfptr);							612

Figure 9-9 (Part 9 of 11). Source Program Example — CSRCDMUL (User-Defined Formats)

```

613 | } | 613
614 | | 614
615 | | 615
616 | /*-----*/ | 616
617 | /* Print Message */ | 617
618 | /* Write message and return code to print file. */ | 618
619 | /*-----*/ | 619
620 | 25 | 620
621 | void print_msg(int mtype) | 621
622 | { | 622
623 | 1 | get_access_to_fb(); | 623
624 | 2 | strncpy(print_rec.major, dsp_icf_fdbk->major_ret_code, 2); | 624
625 | 3 | strncpy(print_rec.minor, dsp_icf_fdbk->minor_ret_code, 2); | 625
626 | 4 | strncpy(print_rec.filler, " ", 32); | 626
627 | 5 | _Rwrite(prtfptr, "RETURN CODE: ", 36); | 627
628 | 6 | _Rwrite(prtfptr, &print_rec, sizeof(print_rec)); | 628
629 | 7 | if (mtype == NORM_END) | 629
630 | 8 | _Rwrite(prtfptr, "PROGRAM CSRCDMUL COMPLETED NORMALLY ", 36); | 630
631 | | else | 631
632 | 9 | if (mtype == RCD_ERR) | 632
633 | 10 | _Rwrite(prtfptr, "RECORD FORMAT IS INCORRECT ON READ ", 36); | 633
634 | | else | 634
635 | 11 | _Rwrite(prtfptr, "PROGRAM ENDED DUE TO ERROR IN CMNFIL", 36); | 635
636 | | } | 636
637 | | | 637
638 | | | 638
639 | /*-----*/ | 639
640 | /* Get Access to DSP/ICF Feedback */ | 640
641 | /* The feedback areas are updated after each display or ICF file I/O */ | 641
642 | /* operation, and so the pointers must be updated to point to the "new" */ | 642
643 | /* feedback areas to get the return code. The offset to the display/ */ | 643
644 | /* ICF feedback area is contained in the common I/O feedback and is */ | 644
645 | /* added to the pointer to the common feedback area to get access to */ | 645
646 | /* display/ICF feedback area. */ | 646
647 | /*-----*/ | 647
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 14
Line STMT SEQNBR INCNO
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9.....
648 | 26 | 648
649 | void get_access_to_fb() | 649
650 | { | 650
651 | 1 | comm_fdbk = _Riofbk(icffptr); | 651
652 | | dsp_icf_fdbk = (XXIOFB_DSP_ICF_T *)((char *)comm_fdbk + | 652
653 | 2 | comm_fdbk->file_dep_fb_offset); | 653
654 | | } | 654
***** END OF SOURCE *****
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 15
***** INCLUDES *****
INCNO Include Name Actual Include Name
1 dspf CICFLIB/dspfil(*all)
2 icff/itmrspl CICFLIB/cmnfil(itmrspl)
3 icff/dtlrspl CICFLIB/cmnfil(dtlrspl)
4 icff/itmreql CICFLIB/cmnfil(itmreql)
5 icff/dtlreql CICFLIB/cmnfil(dtlreql)
6 stdio.h QCC/H/STDIO
7 stddef.h QCC/H/STDDEF
8 errno.h QCC/H/ERRNO
9 signal.h QCC/H/SIGNAL
10 ctype.h QCC/H/CTYPE
11 stdarg.h QCC/H/STDARG
12 recio.h QCC/H/RECIO
13 xxasio.h QCC/H/XXASIO
14 xxfdbk.h QCC/H/XXFDBK
15 stdlib.h QCC/H/STDLIB
16 stddef.h QCC/H/STDDEF
17 string.h QCC/H/STRING
18 xxfdbk.h QCC/H/XXFDBK
19 xxasio.h QCC/H/XXASIO
20 xxcvt.h QCC/H/XXCVT
***** END OF INCLUDES *****
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CSRCDMUL RCHAS400 08/27/90 14:30:09 Page 16
***** MESSAGE SUMMARY *****
Total Info(0-4) Warning(5-19) Error(20-29) Severe(30-39) Terminal(40-99)
0 0 0 0 0 0
***** END OF MESSAGE SUMMARY *****

```

Figure 9-9 (Part 10 of 11). Source Program Example — CSRCDMUL (User-Defined Formats)

ROUTINE	BLOCK	NUMBER	SCOPE	TYPE
<MAIN>	2		LOCAL	MAIN-PROGRAM
QXXPGMDEV	86		LOCAL	PROCEDURE
QXXACQUIRE	87		LOCAL	PROCEDURE
QXXRELEASE	88		LOCAL	PROCEDURE
QXXFORMAT	89		LOCAL	PROCEDURE
QXXINDARA	90		LOCAL	PROCEDURE
QXXREADINVDEV	91		LOCAL	PROCEDURE
__reads	115		LOCAL	PROCEDURE
__rwrite	118		LOCAL	PROCEDURE
__rfmt	125		LOCAL	PROCEDURE
__memset	172		LOCAL	PROCEDURE
__strncmp	190		LOCAL	PROCEDURE
__strncpy	192		LOCAL	PROCEDURE
QXXDZTOZ	204		LOCAL	PROCEDURE
QXXZTOD	209		LOCAL	PROCEDURE
QXXZTOI	210		LOCAL	PROCEDURE
evoke_target	229		ENTRY	PROCEDURE
process_item_req	230		ENTRY	PROCEDURE
send_item_req	231		ENTRY	PROCEDURE
rec_item_info	232		ENTRY	PROCEDURE
calc_profit_loss	233		ENTRY	PROCEDURE
process_cust_req	234		ENTRY	PROCEDURE
send_cust_req	235		ENTRY	PROCEDURE
rec_cust_info	236		ENTRY	PROCEDURE
pos_ret_code	237		ENTRY	PROCEDURE
check_no_data	238		ENTRY	PROCEDURE
check_timeout	239		ENTRY	PROCEDURE
detach	240		ENTRY	PROCEDURE
end_job	241		ENTRY	PROCEDURE
close_files	242		ENTRY	PROCEDURE
print_msg	243		ENTRY	PROCEDURE
get_access_to_fb	244		ENTRY	PROCEDURE
main	245		ENTRY	PROCEDURE

Figure 9-9 (Part 11 of 11). Source Program Example — CSRCDMUL (User-Defined Formats)

Target Program Multiple-Session

Inquiry: The following describes the C/400 target program for multiple-session inquiry program example.

Program Files: The C/400 multiple-session target program uses the following files:

CFILE

An ICF file used to send records to and receive records from the source program. It is done with the file-level INDARA DDS keyword, indicating a separate indicator area.

PFILE

A database file used to retrieve the record for the item requested from the remote system.

QSYSPT

A printer file used to print error messages resulting from communications errors.

DDS Source: The DDS for the ICF file (CFILE) is illustrated below.

```

*****
A*                                     *
A*                                     ICF FILE                               *
A*                                     USED IN TARGET MULTIPLE SESSION PROGRAM *
A*                                     *
*****
A                                     INDARA
A 05                                  RQSWRT
A 10                                  ALMWRRT
A                                     INDXTX(10 '10 END TRANS.')
A 15                                  EOS
A 20                                  FAIL
A                                     INDXTX(20 '20 F ABORT ST')
A                                     RCVFAIL(25 'RECEIVED FAIL')
A 30                                  DETACH
A                                     INDXTX(30 '30>DETACH TGT')
A                                     RCVDETACH(44 'RCV DETACH')
A                                     RCVTRNRND(40 'END OF TRN')
A                                     R SNDPART
A                                     INVITE
A                                     RECTYP      1
A                                     ITEMNO      6
A                                     EDATA        130
A                                     FILL1        13
A                                     R RCVPART
A                                     RECID2       6

```

The command needed to create the ICF file is:

```

CRTICFF FILE(CICFLIB/CFILE) SRCFILE(CICFLIB/QICFPUB)
SRCMBR(CFILE) ACQPGHDEV(*NONE) TEXT("TARGET ICF FILE
FOR MULTIPLE SESSION PROGRAM")

```

The command needed to define the program device entry is:

```

OVRICFDEVE PGMDEV(ICF00) RMTLOCNAME(*REQUESTER)

```

The DDS source for the database file (PFILE) is illustrated below:

```

5714PW1 R01M00 910524          SEU SOURCE LISTING          10/16/87 07:43:14          PAGE 1
SOURCE FILE . . . . . QICFPUB/ICFLIB
MEMBER . . . . . PFILE
SEQNBR*... 1 ... 2 ... 3 ... 4 ... 5 ... 6 ... 7 ... 8 ... 9 ... 0
100      A                                LIFO                                07/02/87
200      A      R DBREC                                05/06/87
300      A      RECCUS      1                                10/01/87
400      A      DBSEQ      6                                08/18/87
500      A      DBDATA     130                              07/02/87
600      A      DBFILL     13                                10/01/87
700      A      K DBSEQ                                07/04/87
          * * * * * E N D   O F   S O U R C E   * * * * *

```

Program Explanation: The following explains the structure of the program example illustrated in Figure 9-10 on page 9-25. The ICF file used is defined by the user, and uses externally described data formats (DDS). The reference numbers in the explanation below correspond to the numbers in the following program example.

- 1** External file descriptions are included for the ICF file (CFILE).
CFILE is the ICF file used to send records to and receive records from the remote system. CFILE is implemented with the file-level keyword, INDARA, indicating that a separate indicator area is used.
- 2** This structure is used to print the return code to the print file QSYSPRT. Output to QSYSPRT is by record, so the filler field is used to blank out characters that may have been set by a previous write operation to the print file. See **15** and **16**.
- 3** The variables to be used globally by the program are defined here. The common and display/ICF feedback area pointers, the file pointers, and the ICF file separate indicator area are defined.
- 4** The routines, except the main routine, are prototyped so the compiler knows the type of value returned and the type of parameters passed, if any.
- 5** The ICF, printer, and database files are opened for record input, output, or both. The ICF file is opened with the separate indicator area option specified.
- 6** The separate indicator area is initialized and defined. The variable `icf_indic` is a character array of size 99.
- 7** The program device ICF00 used by the program is explicitly acquired.
- 8** This program continues reading from and writing to the ICF file until either a detach is received from the source program, the

source program ends abnormally, or an error occurs in the transaction.

A call to a procedure (section **13**) to close the files is made before the program ends.

- 9** This function reads data from the ICF file (CFILE) that was sent from the remote system.
A check is made to see if a detach was received (indicator 44 in CFILE) from the remote system, in which case control returns to **8** and the program ends. Note that the subscript to the indicator in the separate indicator area array is offset by one since the first indicator starts at position zero in the array.
If a turnaround indication is received (indicator 40 in CFILE), then the return code is checked for a value of 3431 (not a serious error) or a 0000. Any other return code received causes an error message to be printed and the program to end upon return to **8**. If the turnaround is not received, the program ends.
- 10** The number received from the remote system is used to find the corresponding item or customer record by key in PFILE. If the record is not found, 000000 is sent to the source program. The record ID is set to 'I'.
The data is sent to the remote system with an invite, and the return code is checked for a successful operation. If the major return code is not 00, the program ends.
- 11** This function calls a procedure (section **16**) to access the display/ICF feedback area. It then checks for a 3431 return code or a 00 major return code.
- 12** This function calls a procedure (section **16**) to access the display/ICF feedback area. It then checks for a 00 major return code, which indicates that the last operation was successful.

- 13** This procedure closes the ICF, printer, and database files.
- 14** This procedure retrieves the return code to be printed from the display/ICF feedback area and prints a normal end message to the print file.
- 15** This procedure retrieves the return code to be printed from the display/ICF feedback area and prints an abnormal end message to the print file.

- 16** This procedure accesses the common I/O feedback and the display/ICF feedback areas. The pointers must be reset after each I/O operation where information from the feedback areas is needed, so this procedure is called before checking the return code.

```

5738CB1 V2R1M0 910524 IBM AS/400 C/400          CICFLIB/CTGTDMUL          RCHAS400          09/20/90 14:46:32  Page 1
          * * * * * P R O L O G   * * * * *

Program name . . . . . : CTGTDMUL
Library name . . . . . : CICFLIB
Source file . . . . . : QICFPUB
Library name . . . . . : CICFLIB
Source member name . . . . . : CTGTDMUL
Text Description . . . . . : Target C program for ICF Programmer's Guide
Compiler options . . . . . : *SOURCE *NOXREF *NOSHOWUSR *NOSHOWSYS *NOSHOWSKP *NOEXPMAC *NOAGR
                          : *NOPPONLY *NODEBUG *GEN *NOSECLVL *PRINT *LOGMSG
Language level options . . . . . : *EXTENDED
Source margins:
Left margin . . . . . : 1
Right margin . . . . . : 32767
Sequence columns:
Left Column . . . . . :
Right Column . . . . . :
Define name . . . . . :
Generation options . . . . . : *NOLIST *NOXREF *GEN *NOATR *NODUMP *NOOPTIMIZE *NOALWBND
                          : *NOANNO
Print file . . . . . : QSYSPRT
Library name . . . . . : *LIBL
Message flagging level . . . . . : 0
Compiler message:
Message limit . . . . . : *NOMAX
Message limit severity . . . . . : 30
Replace program object . . . . . : *YES
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Target Release . . . . . : *CURRENT
Last change . . . . . : 90/09/20 14:46:14
Source description . . . . . : Target C program for ICF Programmer's Guide
Compiler . . . . . : IBM C/400 Compiler
5738CB1 V2R1M0 910524 IBM AS/400 C/400          CICFLIB/CTGTDMUL          RCHAS400          09/20/90 14:46:32  Page 2
          * * * * * S O U R C E   * * * * *

Line STMT          *...+.....1....+.....2....+.....3....+.....4....+.....5....+.....6....+.....7....+.....8....+.....9..... SEQNBR  INCNO
1      | 1 | #pragma mapinc("icff/snd", "cicflib/cfile(sndpart)", "output", "p z") | 1
2      | 2 | #include "icff/snd" | 2
3      | 3 | #pragma mapinc("icff/rcv", "cicflib/cfile(rcvpart)", "input", "p z") | 3
4      | 4 | #include "icff/rcv" | 4
5      | 5 | /*-----*/ | 5
6      | 6 | /* This program will handle the request for either a customer number or */ | 6
7      | 7 | /* an item number. This is accomplished by making the data base file */ | 7
8      | 8 | /* structure (key length, key position, record length, record size, */ | 8
9      | 9 | /* etc.) the same for both files with only the record contents */ | 9
10     | 10 | /* different. The database file is searched with a customer number or */ | 10
11     | 11 | /* item number as a key. */ | 11
12     | 12 | /* This program ends when a detach request is received from the source */ | 12
13     | 13 | /* program. */ | 13
14     | 14 | /* Indicators associated with the ICF file are defined and are */ | 14
15     | 15 | /* referenced for every I/O operation issued. */ | 15
16     | 16 | /*-----*/ | 16
17     | 17 | | 17
18     | 18 | #define ION '1' /* Indicator set on */ | 18
19     | 19 | #define IOFF '0' | 19
20     | 20 | #define ERROR 1 /* Error occurred */ | 20
21     | 21 | #define NOERROR 0 | 21
22     | 22 | #define TRUE 1 | 22
23     | 23 | #define FALSE 0 | 23

```

Figure 9-10 (Part 1 of 6). Target Program Example — CTGTDMUL (User-Defined Formats)

```

24 | #include <stdio.h> /* Standard I/O header */ 24
25 | #include <recio.h> /* Record I/O header */ 25
26 | #include <stdlib.h> /* General utilities */ 26
27 | #include <stddef.h> /* Standard definitions */ 27
28 | #include <string.h> /* String handling utilities */ 28
29 | #include <xxfdbk.h> /* Feedback area structures */ 29
30 | #include <xxasio.h> /* Indicator area structure */ 30
31 | | 2 31
32 | /*-----*/ 32
33 | /* Define the structure to be used to write to the print file. */ 33
34 | /*-----*/ 34
35 | 35
36 | struct { 36
37 |     char major??(???); 37
38 |     char minor??(???); 38
39 |     char filler??(29??); /* Used for padding with blanks */ 39
40 | } print_rec; 40
41 | | 3 41
42 | /*-----*/ 42
43 | /* Declare global variables. */ 43
44 | /*-----*/ 44
45 | 45
46 | _RFILE *icffptr; /* Ptr to ICF file */ 46
47 | _RFILE *prtfptr; /* Ptr to print file */ 47
48 | _RFILE *dbfptr; /* Ptr to database file */ 48
49 | _RIOFB_T *rio_fdbk; /* Ptr to partial I/O feedback */ 49
50 | _XXIOFB_T *comm_fdbk; /* Ptr to common I/O feedback */ 50
51 | _XXIOFB_DSP_ICF_T *dsp_icf_fdbk; /* Ptr to dsp/ICF I/O feedback */ 51
52 | char icf_indic??(99??); /* ICF file separate indic area */ 52
5738CB1 V2RIM0 910524 IBM AS/400 C/400 CICFLIB/CTGTDML RCHAS400 09/20/90 14:46:32 Page 3
Line STMT SEONBR INCNO
53 | *...1...2...3...4...5...6...7...8...9..... 53
54 | |CICFLIB_CFILE_SNDPART_o_t sndpart_icf_o; 54
55 | |CICFLIB_CFILE_RCVPART_i_t rcvpart_icf_i; 55
56 | | 4 56
57 | |int read_cfile(void); 57
58 | |int send_data(void); 58
59 | |int check_ret_code(void); 59
60 | |int pos_ret_code(void); 60
61 | |void close_files(void); 61
62 | |void print_norm_end(void); 62
63 | |void print_error_end(void); 63
64 | |void get_access_to_fb(void); 64
65 | 65
66 | |main() 66
67 | |{ 67
68 | |    int error_or_end = FALSE; /* Set if I/O error occurred */ 68
69 | |    | 5 69
70 | |    /*-----*/ 70
71 | |    /* Open the ICF and printer files. If an error occurs the */ 71
72 | |    /* program ends. */ 72
73 | |    /*-----*/ 73
74 | | 74
75 | |    if ((icffptr = _Ropen("CICFLIB/CFILE", "ar+ indicators=y riofb=y")) 75
76 | |        == NULL) 76
77 | |        exit(ERROR); 77
78 | |    if ((prtfptr = _Ropen("CICFLIB/QSYSPRT", "wr")) == NULL) { 78
79 | |        _Rclose(icffptr); 79
80 | |        exit(ERROR); 80
81 | |    } 81
82 | | 82
83 | |    if ((dbfptr = _Ropen("CICFLIB/PFILE", "rr riofb=y")) == NULL) { 83
84 | |        _Rclose(icffptr); 84
85 | |        _Rclose(prtfptr); 85
86 | |        exit(ERROR); 86
87 | |    } 87
88 | |    | 6 88
89 | |    /*-----*/ 89
90 | |    /* Set up separate indicator area for the ICF file. */ 90
91 | |    /*-----*/ 91
92 | | 92
93 | |    memset(icf_indic, IOFF, 99); 93
94 | |    _Rindara(icffptr, icf_indic); 94
95 | |    | 7 95
96 | |    /*-----*/ 96
97 | |    /* Explicitly acquire a session. If an error occurs the */ 97
98 | |    /* files will be closed and the program will end. */ 98
99 | |    /*-----*/ 99
100 | 100

```

Figure 9-10 (Part 2 of 6). Target Program Example — CTGTDML (User-Defined Formats)

```

101 12 |   _Racquire(icffptr, "ICF00");
102 13 |   if (pos_ret_code() == ERROR) {
103 14 |       print_error_end();
104 15 |       close_files();
105 16 |       exit(ERROR);
106 16 |   }
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CTGTMUL RCHAS400 09/20/90 14:46:32 Page 4
Line STMT SEQNBR INCNO
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9...
107 8 |
108 | /*-----*/
109 | /* Process requests from the source program until a detach */
110 | /* is received or an error occurs. */
111 | /*-----*/
112 |
113 17 |   _Rpgmdev(icffptr, "ICF00");
114 18 |   while (error_or_end == FALSE) {
115 19 |       if (read_cfile() == NOERROR) {
116 20 |           if (send_data() == ERROR)
117 21 |               error_or_end = TRUE;
118 |       }
119 |       else
120 22 |           error_or_end = TRUE;
121 |   }
122 23 |   close_files();
123 | }
124 |
125 |
126 | /*-----*/
127 | /*           Read ICF File */
128 | /* This routine issues a read operation to the program device. The */
129 | /* detach indication is checked, and if it was signaled main is */
130 | /* notified. If the receive turnaround indicator isn't set, or an */
131 | /* unexpected return code is received, main is notified of an error. */
132 | /*-----*/
133 | 9 |
134 | read_cfile()
135 | {
136 | 1 |   _Rformat(icffptr, "RCVPART");
137 | 2 |   memset(icf_indic, 10FF, 99);
138 | 3 |   _Rreadn(icffptr, &rcvpart_icf_i, sizeof(rcvpart_icf_i), __DFT);
139 | 4 |   if (icf_indic??(43??) == ION) { /* Detach indicator (44) set */
140 | 5 |       print_norm_end();
141 | 6 |       return(ERROR);
142 |   }
143 |   else
144 | 7 |       if (icf_indic??(39??) == ION) /* Rec. turnaround (40) set */
145 | 8 |           if (check_ret_code() == NOERROR)
146 | 9 |               return(NOERROR);
147 |           else {
148 | 10 |               print_error_end();
149 | 11 |               return(ERROR);
150 |           }
151 |       else {
152 | 12 |           print_error_end();
153 | 13 |           return(ERROR);
154 |       }
155 |   }
156 | }
157 |
158 | /*-----*/
159 | /*           Write to ICF File */
160 | /* A request from the source program results in reading a single record */
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CTGTMUL RCHAS400 09/20/90 14:46:32 Page 5
Line STMT SEQNBR INCNO
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9...
161 | /* containing the requested customer or order number. The response will */
162 | /* be returned in a single record containing either the item or customer */
163 | /* information, depending on the data base content. */
164 | /* The response is sent to the source program by writing to the program */
165 | /* device file using format sndpart. */
166 | /* The database file is searched by key. If the number of bytes */
167 | /* returned is 0, then the record was not found on the read, in which */
168 | /* case "000000" is sent back to the source program. */
169 | /*-----*/
170 | 10 |
171 | send_data()
172 | {
173 | 1 |   rio_fdbk = _Rreadk(dbfptr, &sndpart_icf_o, sizeof(sndpart_icf_o),
174 |   _KEY_EQ, rcvpart_icf_i.RECID2, 6);

```

Figure 9-10 (Part 3 of 6). Target Program Example — CTGTMUL (User-Defined Formats)

```

175 2 |   if (rio_fdbk->num_bytes == 0) {
176 3 |       sndpart_icf_o.RECTYP = 'I';
177 4 |       strncpy(sndpart_icf_o.ITEMNO, "000000", 6);
178 |   }
179 5 |   _Rformat(icffptr, "SNDPART");
180 6 |   _Rwrite(icffptr, &sndpart_icf_o, sizeof(sndpart_icf_o));
181 7 |   if (pos_ret_code() == NOERROR)
182 8 |       return(NOERROR);
183 |   else {
184 9 |       print_error_end();
185 10 |       return(ERROR);
186 |   }
187 | }
188 |
189 |
190 | /*-----*/
191 | /*          Check Return Code          */
192 | /* This routine checks the return code after a receive operation for */
193 | /* 0000 and 3431. Anything else is considered an error.          */
194 | /*-----*/
195 | 11
196 | check_ret_code()
197 | {
198 |     get_access_to_fb();
199 |     if (strncmp(dsp_icf_fdbk->major_ret_code, "34", 2) == 0 &&
200 |         strncmp(dsp_icf_fdbk->minor_ret_code, "31", 2) == 0)
201 |         return(NOERROR);
202 |     else
203 |         return(pos_ret_code());
204 | }
205 |
206 |
207 | /*-----*/
208 | /*          Check for Successful Operation          */
209 | /* This routine checks the major return code of 00 to see if the last */
210 | /* operation was successful.          */
211 | /*-----*/
212 | 12
213 | pos_ret_code()
214 | {
5738CB1 V2R1M0 910524 IBM AS/400 C/400      CICFLIB/CTGTDMUL      RCHAS400      09/20/90 14:46:32      Page 6
Line STMT      *...+....1...+....2...+....3...+....4...+....5...+....6...+....7...+....8...+....9.....      SEQNBR      INCRNO
215 1 |   get_access_to_fb();
216 2 |   if (strncmp(dsp_icf_fdbk->major_ret_code, "00", 2) == 0)
217 3 |       return(NOERROR);
218 |   else
219 4 |       return(ERROR);
220 | }
221 |
222 |
223 | /*-----*/
224 | /*          Close the Files          */
225 | /* Close the ICF, print, and database files.          */
226 | /*-----*/
227 | 13
228 | void close_files()
229 | {
230 |     _Rclose(icffptr);
231 |     _Rclose(prtfptr);
232 |     _Rclose(dbfptr);
233 | }
234 |
235 |
236 | /*-----*/
237 | /*          Print Normal End Message          */
238 | /* Write normal end message and return code to print file.          */
239 | /*-----*/
240 | 14
241 | void print_norm_end()
242 | {
243 |     get_access_to_fb();
244 |     strncpy(print_rec.major, dsp_icf_fdbk->major_ret_code, 2);
245 |     strncpy(print_rec.minor, dsp_icf_fdbk->minor_ret_code, 2);
246 |     strncpy(print_rec.filler, "          ", 29);
247 |     _Rwrite(prtfptr, "RETURN CODE:          ", 33);
248 |     _Rwrite(prtfptr, &print_rec, sizeof(print_rec));
249 |     _Rwrite(prtfptr, "CTGTDMUL HAS COMPLETED NORMALLY ", 33);
250 | }
251 |

```

Figure 9-10 (Part 4 of 6). Target Program Example – CTGTDMUL (User-Defined Formats)

```

252 |                                                                           | 252
253 | /*-----*/ | 253
254 | /*          Print Abnormal End Message          */ | 254
255 | /* Write abnormal end message and return code to print file. */ | 255
256 | /*-----*/ | 256
257 | 15 | | 257
258 | void print_error_end() | 258
259 | { | 259
260 | 1 | get_access_to_fb(); | 260
261 | 2 | strncpy(print_rec.major, dsp_icf_fdbk->major_ret_code, 2); | 261
262 | 3 | strncpy(print_rec.minor, dsp_icf_fdbk->minor_ret_code, 2); | 262
263 | 4 | strncpy(print_rec.filler, "          ", 29); | 263
264 | 5 | _Rwrite(prtfptr, "RETURN CODE:          ", 33); | 264
265 | 6 | _Rwrite(prtfptr, &print_rec, sizeof(print_rec)); | 265
266 | 7 | _Rwrite(prtfptr, "CTGTDMUL HAS COMPLETED ABNORMALLY", 33); | 266
267 | } | 267
268 | | 268
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CTGTDMUL RCHAS400 09/20/90 14:46:32 Page 7
Line STMT *...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9..... SEQNBR INCNO

269 |                                                                           | 269
270 | /*-----*/ | 270
271 | /*          Get Access to DSP/ICF Feedback          */ | 271
272 | /* The feedback areas are updated after each display or ICF file I/O */ | 272
273 | /* operation, and so the pointers must be updated to point to the "new" */ | 273
274 | /* feedback areas to get the return code. The offset to the display/ */ | 274
275 | /* ICF feedback area is contained in the common I/O feedback and is */ | 275
276 | /* added to the pointer to the common feedback area to get access to */ | 276
277 | /* display/ICF feedback area. */ | 277
278 | /*-----*/ | 278
279 | 16 | | 279
280 | void get_access_to_fb() | 280
281 | { | 281
282 | 1 | comm_fdbk = _Riofbk(icffptr); | 282
283 | 2 | dsp_icf_fdbk = (XXIOFB_DSP_ICF_T *)((char *)comm_fdbk + | 283
284 |   |   comm_fdbk->file_dep_fb_offset); | 284
285 | } | 285
          * * * * *   E N D   O F   S O U R C E   * * * * *
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CTGTDMUL RCHAS400 09/20/90 14:46:32 Page 8
          * * * * *   I N C L U D E S   * * * * *
INCNO Include Name                               Last change          Actual Include Name
 1 icff/snd                                90/09/12 14:42:23   CICFLIB/cfile(sndpart)
 2 icff/rcv                                90/09/12 14:42:22   CICFLIB/cfile(rcvpart)
 3 stdio.h                                 90/09/12 14:42:23   QCC/H/STDIO
 4 stddef.h                                90/09/12 14:42:22   QCC/H/STDEDEF
 5 errno.h                                  90/09/12 14:42:18   QCC/H/ERRNO
 6 signal.h                                 90/09/12 14:42:22   QCC/H/SIGNAL
 7 ctype.h                                  90/09/12 14:42:18   QCC/H/CTYPE
 8 stdarg.h                                 90/09/12 14:42:22   QCC/H/STDARG
 9 recio.h                                  90/09/12 14:42:21   QCC/H/RECIO
10 xxasio.h                                 90/09/12 14:42:26   QCC/H/XXASIO
11 xxfdbk.h                                 90/09/12 14:42:28   QCC/H/XXFDBK
12 stdlib.h                                 90/09/12 14:42:24   QCC/H/STDLIB
13 stddef.h                                 90/09/12 14:42:22   QCC/H/STDEDEF
14 string.h                                 90/09/12 14:42:24   QCC/H/STRING
15 xxfdbk.h                                 90/09/12 14:42:28   QCC/H/XXFDBK
16 xxasio.h                                 90/09/12 14:42:26   QCC/H/XXASIO
          * * * * *   E N D   O F   I N C L U D E S   * * * * *
5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CTGTDMUL RCHAS400 09/20/90 14:46:32 Page 9
          * * * * *   M E S S A G E   S U M M A R Y   * * * * *
        Total      Info(0-4)      Warning(5-19)      Error(20-29)      Severe(30-39)      Terminal(40-99)
          0          0              0                  0                  0                  0
          * * * * *   E N D   O F   M E S S A G E   S U M M A R Y   * * * * *

```

Figure 9-10 (Part 5 of 6). Target Program Example — CTGTDMUL (User-Defined Formats)

```

5738CB1 V2R1M0 910524 IBM AS/400 C/400 CICFLIB/CTGDMUL RCHAS400 09/20/90 14:46:32 Page 10
ROUTINE      BLOCK NUMBER SCOPE TYPE
<MAIN>       2          LOCAL MAIN-PROGRAM
QXXPGMDEV    86          LOCAL PROCEDURE
QXXACQUIRE  87          LOCAL PROCEDURE
QXXFORMAT    89          LOCAL PROCEDURE
QXXINDARA    90          LOCAL PROCEDURE
__reads      115         LOCAL PROCEDURE
__readk      117         LOCAL PROCEDURE
__rwrite     118         LOCAL PROCEDURE
__rfmt       125         LOCAL PROCEDURE
__memset     172         LOCAL PROCEDURE
__strncmp    190         LOCAL PROCEDURE
__strncpy    192         LOCAL PROCEDURE
read_cfile   213         ENTRY  PROCEDURE
send_data    214         ENTRY  PROCEDURE
check_ret_code 215         ENTRY  PROCEDURE
pos_ret_code 216         ENTRY  PROCEDURE
close_files  217         ENTRY  PROCEDURE
print_norm_end 218         ENTRY  PROCEDURE
print_error_end 219         ENTRY  PROCEDURE
get_access_to_fb 220         ENTRY  PROCEDURE
main         221         ENTRY  PROCEDURE
          ***** END OF COMPILATION *****

```

Figure 9-10 (Part 6 of 6). Target Program Example — CTGDMUL (User-Defined Formats)

Chapter 10. Communications Applications with COBOL/400 Programming Language

Previous chapters in this manual describe the functions provided by ICF. This chapter introduces you to the COBOL/400 interfaces for ICF and provides program examples.

Two application examples are presented in this chapter. For each example, both the source and target programs are provided. Each program is written first with user-defined formats (data description specifications, DDS) and then with system-supplied formats.

The first example in this section is a batch data transfer application using a single session. The second example is a multiple-session inquiry application using one display file and four ICF sessions.

Not all programming considerations or techniques are illustrated in each example in this section. Review these examples and the examples provided in the appropriate programming manual before beginning application design and coding.

Note: The examples in this section were written to the APPC communications type. Minor changes might be required if another communications type is used.

Introduction to the COBOL/400 Interface

Before you write a COBOL/400 communications application, you must understand the high-level language interface provided by COBOL/400 programming language.

The operations you use in the communications portion of your program are similar to work station operations. ICF files are defined as transaction files in COBOL/400 programming language. In the noncommunications portion of your program, you can use all noncommunications operations you normally use to process data that is sent or received between your program and the remote program.

Figure 10-1 briefly introduces the COBOL/400 statements you use in the communications portion of your program.

Figure 10-1. COBOL/400 Statements

ICF Operation	COBOL/400 Statement	Function
Open	OPEN	Opens the ICF file
Acquire	ACQUIRE	Establishes a session
Get-Attributes	ACCEPT	Gets the attributes of a session
Read	READ ¹	Receives data from a specific program device
Read-from-invited-program-devices	READ ¹	Receives data from any invited program device ²
Write	WRITE	Performs many of the ICF communications functions within a session
Release	DROP	Releases the session
Close	CLOSE	Closes the ICF file

- ¹ A COBOL/400 read operation can be directed either to a specific program device or to all invited program devices. The support provided by the COBOL/400 compiler determines whether to issue an ICF read or read-from-invited-program-devices operation based on the presence of a format name or a terminal name on the read operation. For example, if a READ is sent with a specific format or terminal specified, the read operation is interpreted as an ICF read operation. Refer to the COBOL/400 language manual for more information.
- ² The read-from-invited-program-devices operation could complete without data if the timer interval established with either the timer function or WAITRCD expires, or your job is ended (controlled).

Refer to *COBOL/400* User's Guide* for details on the syntax and function of each operation.

Error Handling: The FILE STATUS clause of the FILE-CONTROL paragraph is used in COBOL/400 programming to specify the variables for the COBOL file status and the ICF major and minor return codes.

Figure 10-2 shows the file status values as returned by COBOL/400 after an input/output (I/O) operation for each major and minor return code. Use this list to determine the ICF return code or group of codes that corresponds to the file status value.

Figure 10-2. File Status Values for Major and Minor Return Codes

ICF Return Code	COBOL/400 Return Code
00xx	00
02xx	9A
03xx	00
0309	9A
04xx	9I
0800	00
1100	10
34xx	9G
80xx	30
81xx	92
82xx	9C
83xx	9N
83E0	9K

Accessing the Feedback Areas: Use the COBOL/400 ACCEPT statement to obtain the open or I/O feedback information for an ICF file.

The COBOL/400 offset values for the open and I/O feedback areas are the same as those listed in Appendix C.

Example Programs

The programs presented in this section are:

- Example I (Batch Data Transfer)

Figure 10-3 on page 10-3 shows a batch data transfer program that reads a database file and sends the data to a remote system. When the source program finishes sending its records, it sends an indication that it is done sending records to the target program. The target program then starts sending its records until it reaches end-of-file. At end-of-file, the target program sends a detach indication to the source program. The two programs end their sessions.

- Example II (Multiple-Session Inquiry)

Figure 10-4 on page 10-3 shows an inquiry program that accepts inquiries from a display device, sends the request to one of four remote AS/400 systems, and waits for a response to the inquiry. Based on the input received from the display device, the program determines the target program to which it sends the inquiry request. The same program resides in each of the remote systems.

Figure 10-4 on page 10-3 contains a display device and four ICF communication program devices.

The rest of this chapter discusses the details of the two application examples. The DDS file, program listings, and an explanation of the programs are included.

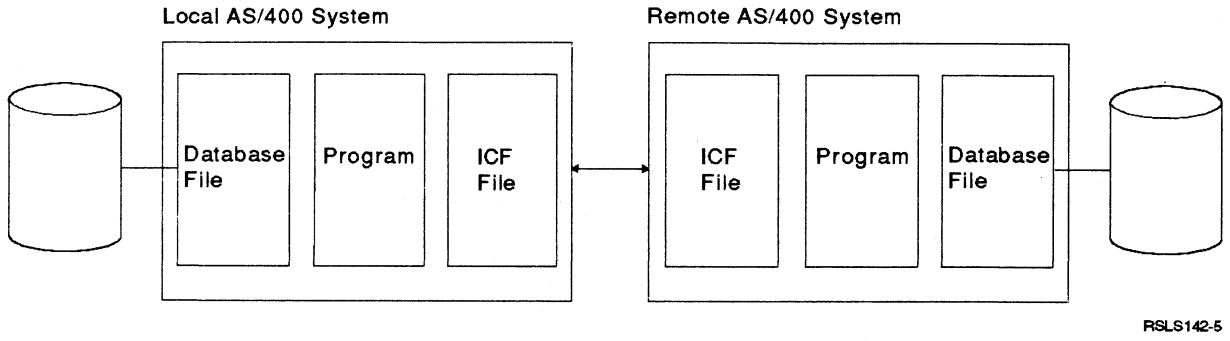


Figure 10-3. Batch Data Transfer

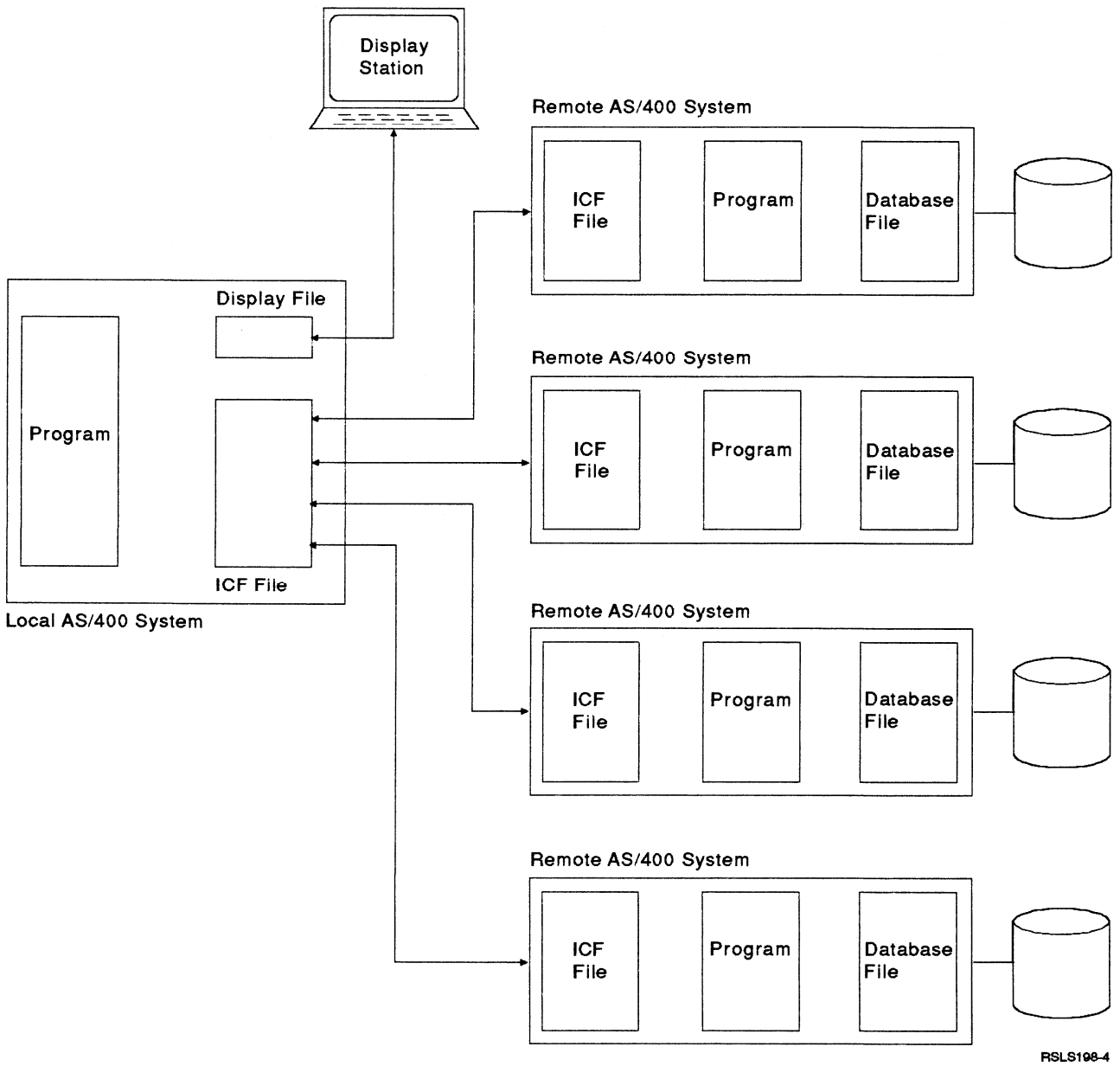


Figure 10-4. Multiple-Session Inquiry

Batch Data Transfer (Example I)

The following figures show a batch data transfer program. A source AS/400 system program communicates with a target program on another AS/400 system using the ICF support. The source program starts a target program on a remote AS/400 system and transfers a file to that target program.

The target program responds, after receiving an indication that the source is done sending, by reading its own file and then sends the records to the source program until it reaches end of file. At end-of-file, the target program sends a detach

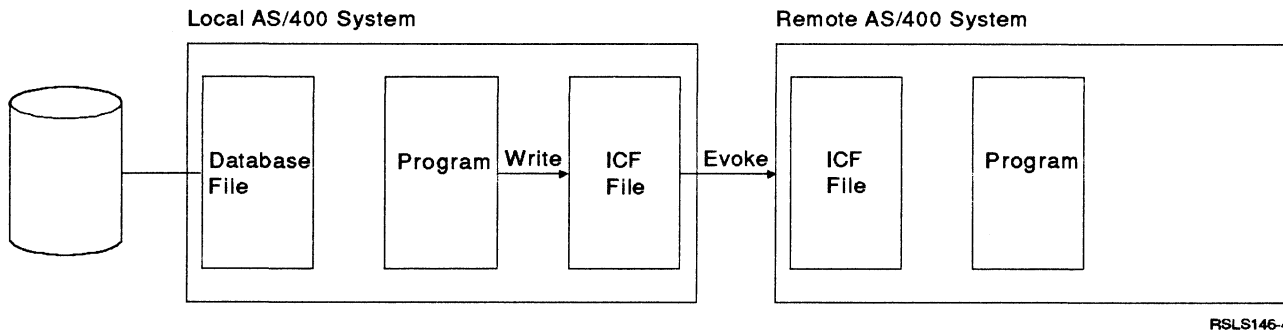
request to the source program and ends its session.

Both the source program and the target program are described.

Transaction Flow of the Batch Data

Transfer (Example I): In Figure 10-5, the source program issues an evoke to start a program at the remote AS/400 system.

Note: An acquire operation is not necessary since the device was acquired during the open operation. The device was acquired during the open operation because the ACQPGMDEV parameter was used when the ICF file was created.

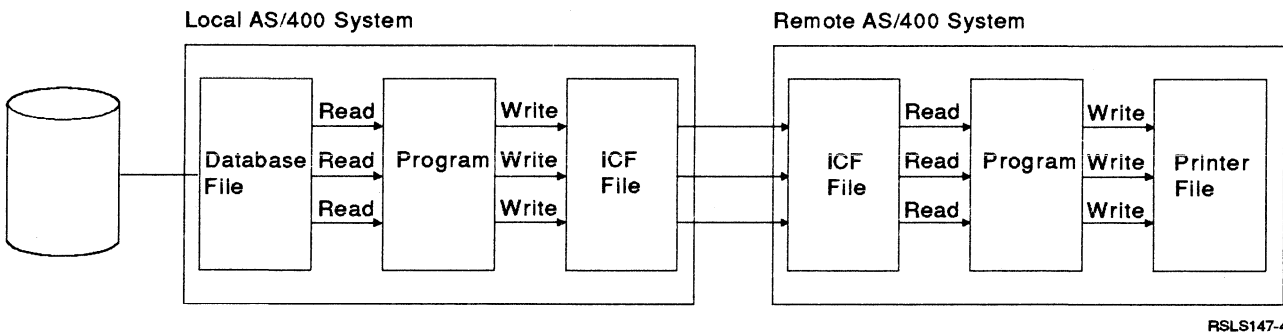


RSL5146-4

Figure 10-5. Evoke Request Starts a Target Program

After issuing the evoke request, the source program sends a database file to the target

program, which prints the records as shown in Figure 10-6.

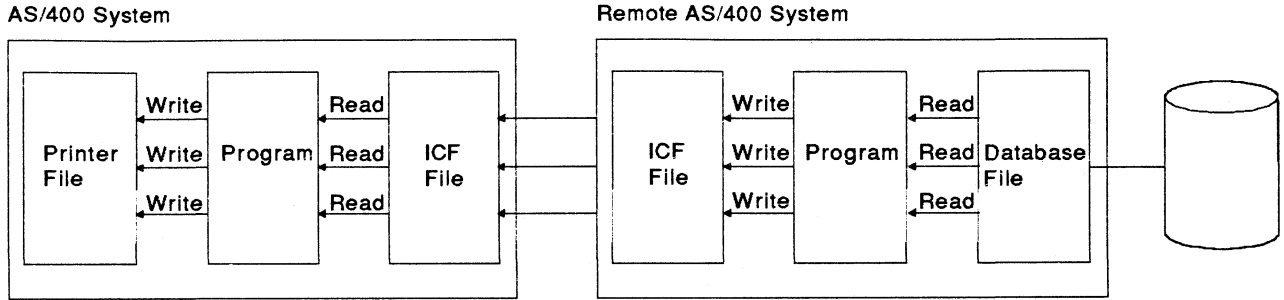


RSL5147-4

Figure 10-6. Target Program Prints Records

After the target program receives and prints the file, a database file is sent to the source program. The source program prints the

records as they are received, as shown in Figure 10-7 on page 10-5.

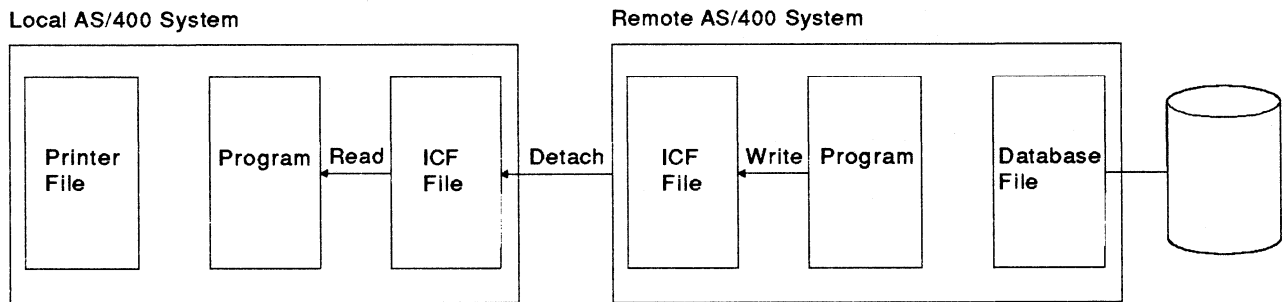


RSL148-4

Figure 10-7. Source Program Prints the Received Records

Once all the records have been sent by the target program, the target program issues a

detach to the source program to end the transaction, as shown in Figure 10-8.



RSL148-4

Figure 10-8. Target Program Ends the Transaction

Source Program Batch Transfer

(Example 1): The following describes the COBOL/400 batch data transfer source program.

Program Files: The COBOL/400 batch transfer source program uses the following files:

SRCICF

An ICF file used to send records to and receive records from the target program.

DBFILE

A database file that contains the records to be sent to the target program.

QPRINT

A printer file used to print the records received from the target program.

DDS Source: The DDS source used in the ICF file is illustrated in the following example. The other files (DBFILE and QPRINT) are program-described and therefore require no DDS.

```

A*****
A*                                     *
A*                               ICF FILE *
A*          USED IN BATCH DATA TRANSFER *
A*                                     *
A*****
A* FILE LEVEL INDICATORS:
A*
A*                               INDARA
A*                               RCVTRNRND(15 'END OF DATA')
A*
A 30                               DETACH
A*
A*                               INDXTX(30 '30--DETACH TARG-
A*                               ET PROGRAM.')
A*
A*                               RCVDATCH(35 'RECEIVED
A*                               DETACHED.')
A*
A*
A*****
A*                               ICF RECORD FORMATS *
A*****
A*                               R RCVDATA
A*                               RCVFLD      BBA
A*                               R SNDDATA
A*                               SNDFLD      BBA
A*                               R EVOKPGM
A*
A 50                               EVOKE(&LIB/&PGMID);
A 50                               SECURITY(2 'PASSWRD' +
A*                               3 'USERID')
A*
A*                               PGMID      18A P
A*                               LIB        18A P
A*                               R ENDREC
A*                               R INVITE
A 45                               INVITE

```

ICF File Creation and Program Device Entry

Definition: The command needed to create the ICF file is:

```
CRTICFF FILE(ICFLIB/SRCICF) SRCFILE(ICFLIB/QICFPUB)
SRCHBR(SRCICF) ACOPGMDEV(PGMDEVA) TEXT('ICF FILE FOR
BATCH DATA TRANSFER')
```

The command needed to define the program device entry is:

```
ADDICFDEVE FILE(ICFLIB/SRCICF) PGMDEV(PGMDEVA)
RMTLOCNAME(CHICAGO)
```

Program Explanation: The following describes the structure of the program examples illustrated in Figure 10-9 on page 10-8 and Figure 10-10 on page 10-13. The ICF file used in the first example is defined by the user, and uses externally described data formats (DDS). The second example uses the same file, but uses program-described data and system-supplied formats. The reference numbers in the explanation below correspond to the numbers in the following program examples.

Although the basic structure of the two examples provided is the same, there are differences because of the use of user-defined formats and system-supplied formats.

Differences between the first and second example are documented as notes in each of the descriptions.

1 This section identifies the files used in the program. SRCICF is the ICF file used to send records to the target program.

This section also contains declarations of I/O variables, work areas and constants needed. MAJ-MIN contains the major/minor return code from the I/O feedback area.

Note: In the program using system-supplied formats, the input records for SRCICF are explicitly coded since SRCICF is now treated as a program-described file. The system-supplied file, QICDMF, can be used instead of SRCICF. This can be done by specifying QICDMF in the file specification, or by using an override ICF file (OVRICFF) command to change the file name from SRCICF to QICDMF. The OVRICFF command can also be used to change the ACQPGMDEV parameter of the file.

2 This section defines the error handling for the program. The major/minor return code is checked to determine whether the error is recoverable. If the error is recoverable

(major/minor code 83xx or 03xx), it sets a flag (ERR-SW) to 1 and returns to the program.

If any other error has occurred, the program writes the feedback data to a file (DFILE), calls a program to print DFILE, and then ends.

3 The program opens the files that are going to be used as follows:

DBFILE

The database file used as input for transmitting to the target program.

QPRINT

The printer file used as output for records received

SRCICF

The ICF file used to send data to and receive data from the target program

Because the ICF file was created using the ACQPGMDEV parameter, the program device is automatically acquired when the file is opened. Therefore, no acquire operation is coded in the program.

4 If the ERR-SW flag is set to 1, indicating that a recoverable error has occurred, the program determines whether the open retry count limit (9) has been exceeded. If it has, the program goes to **11**. If the limit count is less than 9, one is added to the count and control passes to **10**. Control then passes to **3** to attempt to open the file.

5 The evoke request is built and sent to the remote system. Because the DDS for the record format only specifies the field identifiers with the record, the code in this section of the program moves the literal value CTDBATCL to the field *PGMID*, and ICFLIB to the field *L/B*. Indicator 50 is set to indicate that the program start request is to be sent.

When the program start request is received at the remote system, ICFLIB is searched for CTDBATCL and that program is then started. CTDBATCL is a control language (CL) program that contains the following statements:

```
ADDLIBLE ICFLIB
CALL ICFLIB/CTDBAT
```

Note: In the program using system-supplied formats, the library and program (ICFLIB/CTFBATCL) are specified as part of the \$\$EVOKNI format. CTFBATCL is a CL program that contains the following statements:

```
ADDLIB ICFLIB  
CALL ICFLIB/CTFBAT
```

- 6** Data is read from the database file. If the record read is end-of-file, the program sets the EOF-DBFILE-SW, performs routine **12** to invite the program device, and goes to **7**.

If it is not the last record, the data is moved to the output buffer and the program goes to **9** to write the record to the program device. When control returns from **9**, the process is repeated (DBFILE read) until end-of-file.

- 7** Data is read from the ICF file (SRCICF). If the operation was successful, and data was received (major return code = '03'), then the data is written to the printer file (QPRINT). If an error occurs on the read, control passes to **10** to attempt recovery. When control returns from **10**, control passes to **3** to attempt to open the files.

Data is read until the detach indication is received from the target program. When detach is received, indicator 35 is set on, as defined by the RCVDETACH keyword in the DDS for the ICF file. Note that RCVDETACH is a file-level keyword.

Note: In the program using system-supplied formats, the minor return code of '08' is checked to verify whether detach is received.

- 8** After the detach request has been received, this routine writes the following message to the print file:

```
CSDBAT HAS COMPLETED NORMALLY
```

The files are closed. The program device is automatically released as a result of the close operation.

Note: In the program using system-supplied formats, the program name is CSFBAT.

- 9** This routine is performed to write data to the ICF file using the format SNDDATA. If an error occurs on the write, control passes to **10** and then finally to **3**.

Note: In the program using system-supplied formats, the \$\$SENDNI format is used instead of the user-defined SNDDATA format.

- 10** This routine is performed when a recoverable session error has occurred. It closes the files (SRCICF, QPRINT and DBFILE) and sets the error switch (ERR-SW) to 0. Control then returns to the statement immediately following the PERFORM statement that passed control to this routine.

- 11** This routine is performed when the application receives a major/minor return code of 03xx or 83xx after it has already attempted to recover 9 times from session errors that caused the error. The program is designed to tolerate only nine failures. It writes the following message to the print file:

```
CSDBAT HAS COMPLETED ABNORMALLY
```

The session is ended.

Note: In the program using system-supplied formats, the program name is CSFBAT.

- 12** This routine is performed to issue an invite request to the program device using format INVITE. If an error occurs, control passes to **10** and then finally to **3**.

Note: In the program using system-supplied formats, the \$\$SEND format is used instead of the user-defined INVITE format.

```

5738CB1 V2R1M0 910524      IBM AS/400 COBOL/400      ICFLIB/CSDBAT      RCHAS400 10/03/90 14:38:26      Page 1
Program . . . . . : CSDBAT
Library . . . . . : ICFLIB
Source file . . . . . : QICFPUB
Library . . . . . : ICFLIB
Source member . . . . . : CSDBAT 02/28/89 13:51:38
Generation severity level . . . . . : 29
Text 'description' . . . . . : coh01 batch file transfer using dds (source)
Source listing options . . . . . : *SOURCE
Generation options . . . . . : *NONE
Message limit:
Number of messages . . . . . : *NOMAX
Message limit severity . . . . . : 29
Print file . . . . . : QSYSVRT
Library . . . . . : *LIBL
FIPS flagging . . . . . : *NOFIPS *NOSEG *NODEB *NOOBSOLETE
SAA flagging . . . . . : *NOFLAG
Flagging severity . . . . . : 0
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Compiler . . . . . : IBM AS/400 COBOL/400

```

```

5738CB1 V2R1M0 910524      AS/400 COBOL Source      ICFLIB/CSDBAT      RCHAS400 10/03/90 14:38:26      Page 2
STMT SEQNBR -A 1 B...2...3...4...5...6...7..IDENTFCN S COPYNAME  CHG DATE
1 000100 IDENTIFICATION DIVISION.                                02/21/89
2 000200 PROGRAM-ID. CSDBAT.                                     02/21/89
000300*****
000400* THIS IS A BATCH FILE TRANSFER PROGRAM THAT READS A SEQUENTIAL * 02/21/89
000500* FILE AND SENDS THE RECORDS TO THE REMOTE SYSTEM UNTIL THE END * 02/21/89
000600* OF FILE IS REACHED. AT THIS TIME, THE PROGRAM STOPS SENDING * 02/21/89
000700* AND STARTS RECEIVING RECORDS FROM THE REMOTE SYSTEM UNTIL * 02/21/89
000800* A DETACH INDICATION IS RECEIVED. * 02/21/89
000900*****
3 001000 ENVIRONMENT DIVISION.                                10/16/87
4 001100 CONFIGURATION SECTION.                               10/16/87
5 001200 SOURCE-COMPUTER. IBM-AS400.                          02/21/89
6 001300 OBJECT-COMPUTER. IBM-AS400.                          02/21/89
7 001400 SPECIAL-NAMES. I-O-FEEDBACK IS FEEDBACK-AREA        10/16/87
8 001500 OPEN-FEEDBACK IS OPEN-FBA.                            10/16/87
9 001600 INPUT-OUTPUT SECTION.                                10/16/87
10 001700 FILE-CONTROL.                                       10/16/87
001800* 1
11 001900 SELECT DBFILE ASSIGN TO DATABASE-DBFILE.             10/16/87
12 002000 SELECT SRCICF ASSIGN TO WORKSTATION-SRCICF-SI        10/16/87
13 002100 ORGANIZATION IS TRANSACTION                          10/16/87
14 002200 FILE STATUS IS STATUS-IND MAJ-MIN.                   10/16/87
15 002300 SELECT DFILE ASSIGN TO DATABASE-HEXDUMP.             10/16/87
16 002400 SELECT QPRINT ASSIGN TO PRINTER-QSYSVRT.             10/16/87
17 002500 DATA DIVISION.                                       10/16/87
18 002600 FILE SECTION.                                         10/16/87
19 002700 FD DBFILE                                           10/16/87
20 002800 LABEL RECORDS ARE STANDARD.                           10/16/87
21 002900 01 DBREC.                                           10/16/87
22 003000 05 DBREC-DATA PIC X(80).                             10/16/87
23 003100 FD SRCICF                                           10/16/87
24 003200 LABEL RECORDS ARE STANDARD.                           10/16/87
25 003300 01 ICFREC.                                           10/16/87
26 003400 COPY DDS-ALL-FORMATS OF SRCICF.                       10/16/87
27 +000001 05 SRCICF-RECORD PIC X(80). <-ALL-FMTS
+000002* INPUT FORMAT:RCVDATA FROM FILE SRCICF OF LIBRARY ICFLIB <-ALL-FMTS
+000003* <-ALL-FMTS
28 +000004 05 RCVDATA-I REDEFINES SRCICF-RECORD. <-ALL-FMTS
29 +000005 06 RCVFLD PIC X(80). <-ALL-FMTS
+000006* OUTPUT FORMAT:RCVDATA FROM FILE SRCICF OF LIBRARY ICFLIB <-ALL-FMTS
+000007* <-ALL-FMTS
30 +000008 05 RCVDATA-0 REDEFINES SRCICF-RECORD. <-ALL-FMTS
31 +000009 06 RCVFLD PIC X(80). <-ALL-FMTS
+000010* INPUT FORMAT:SNDDATA FROM FILE SRCICF OF LIBRARY ICFLIB <-ALL-FMTS
+000011* <-ALL-FMTS
32 +000012 05 SNDDATA-I REDEFINES SRCICF-RECORD. <-ALL-FMTS
33 +000013 06 SNDFLD PIC X(80). <-ALL-FMTS
+000014* OUTPUT FORMAT:SNDDATA FROM FILE SRCICF OF LIBRARY ICFLIB <-ALL-FMTS
+000015* <-ALL-FMTS
34 +000016 05 SNDDATA-0 REDEFINES SRCICF-RECORD. <-ALL-FMTS
35 +000017 06 SNDFLD PIC X(80). <-ALL-FMTS
+000018* INPUT FORMAT:EVOKPGM FROM FILE SRCICF OF LIBRARY ICFLIB <-ALL-FMTS
+000019* <-ALL-FMTS
+000020* 05 EVOKPGM-I REDEFINES SRCICF-RECORD. <-ALL-FMTS
+000021* OUTPUT FORMAT:EVOKPGM FROM FILE SRCICF OF LIBRARY ICFLIB <-ALL-FMTS

```

Figure 10-9 (Part 1 of 5). Source Program Example — CSDBAT (User-Defined Formats)


```

5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDBAT RCHAS400 10/03/90 14:38:26 Page 3
STMT SEONBR -A 1 B.+....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
+000022* <-ALL-FMTS
36 +000023 05 EVOKPGM-0 REDEFINES SRCICF-RECORD. <-ALL-FMTS
37 +000024 06 PGMID PIC X(10). <-ALL-FMTS
38 +000025 06 LIB PIC X(10). <-ALL-FMTS
+000026* INPUT FORMAT:ENDREC FROM FILE SRCICF OF LIBRARY ICFLIB <-ALL-FMTS
+000027* <-ALL-FMTS
+000028* 05 ENDREC-I REDEFINES SRCICF-RECORD. <-ALL-FMTS
+000029* OUTPUT FORMAT:ENDREC FROM FILE SRCICF OF LIBRARY ICFLIB <-ALL-FMTS
+000030* <-ALL-FMTS
+000031* 05 ENDREC-O REDEFINES SRCICF-RECORD. <-ALL-FMTS
+000032* INPUT FORMAT:INVITE FROM FILE SRCICF OF LIBRARY ICFLIB <-ALL-FMTS
+000033* <-ALL-FMTS
+000034* 05 INVITE-I REDEFINES SRCICF-RECORD. <-ALL-FMTS
+000035* OUTPUT FORMAT:INVITE FROM FILE SRCICF OF LIBRARY ICFLIB <-ALL-FMTS
+000036* <-ALL-FMTS
+000037* 05 INVITE-O REDEFINES SRCICF-RECORD. <-ALL-FMTS
39 003500 FD DFILE 10/16/87
40 003600 LABEL RECORDS ARE STANDARD. 10/16/87
41 003700 01 DUMPREC. 10/16/87
42 003800 05 DUMP-MAJ-MIN PIC X(4). 10/16/87
43 003900 05 DUMP-RECORD PIC X(400). 10/16/87
44 004000 FD QPRINT 10/16/87
45 004100 LABEL RECORDS ARE OMITTED. 10/16/87
46 004200 01 PRINTREC PIC X(132). 10/16/87
47 004300 WORKING-STORAGE SECTION. 10/16/87
48 004400 77 STATUS-IND PIC X(2). 10/16/87
49 004500 77 MAJ-MIN-SAV PIC X(4). 10/16/87
50 004600 77 EOF-DBFILE-SW PIC X VALUE "0". 10/16/87
51 004700 77 ERR-SW PIC X VALUE "0". 10/16/87
52 004800 77 INDON PIC 1 VALUE B"1". 10/16/87
53 004900 77 INDOFF PIC 1 VALUE B"0". 10/16/87
54 005000 77 OPEN-COUNT PIC 9(1) VALUE 0. 10/16/87
55 005100 77 LEN PIC 9(10)V9(5) COMP. 10/16/87
005200 10/16/87
56 005300 77 CMD2 PIC X(31) 10/16/87
57 005400 VALUE "CPYF HEXDUMP *LIST PRFTMT(*HEX)". 10/16/87
58 005500 01 CMNF-INDIC-AREA. 10/16/87
59 005600 05 CMNF-INDIC PIC 1 OCCURS 99 TIMES 10/16/87
60 005700 INDICATOR 1. 10/16/87
61 005800 01 OPEN-FBA. 10/16/87
62 005900 05 FILLER PIC X(75). 10/16/87
63 006000 05 RECS-IN-DB PIC 9(09) COMP-4. 10/16/87
64 006100 05 FILLER PIC X(45). 10/16/87
65 006200 01 MAJ-MIN. 10/16/87
66 006300 05 MAJ PIC X(2). 10/16/87
67 006400 05 MIN PIC X(2). 10/16/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDBAT RCHAS400 10/03/90 14:38:26 Page 4
STMT SEONBR -A 1 B.+....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
006500/ 10/16/87
68 006600 PROCEDURE DIVISION. 10/16/87
006700 DECLARATIVES. 10/16/87
006800 ERR-SECTION SECTION. 10/16/87
006900* 2 10/16/87
007000 USE AFTER STANDARD ERROR PROCEDURE ON SRCICF. 10/16/87
007100 SRCICF-EXCEPTION. 10/16/87
007200* 10/16/87
007300* CHECK THE MAJOR/MINOR CODES AND TAKE APPROPRIATE ACTION 10/16/87
007400* 10/16/87
007500* RECOVERABLE SESSION ERROR. CLOSE XPF-ICF FILE. 10/16/87
69 007600 IF MAJ = "03" OR MAJ = "83" 02/21/89
70 007700 MOVE "PROGRAM STARTED AGAIN DUE TO SESSION ERROR" 10/16/87
007800 TO PRINTREC 10/16/87
71 007900 WRITE PRINTREC 10/16/87
72 008000 MOVE "1" TO ERR-SW 10/16/87
73 008100 GO TO EXIT-DECLARATIVES. 10/16/87
008200* 10/16/87
008300***** 02/21/89
008400* WHEN THERE IS A PERMANENT SESSION ERROR DETECTED, THE MAJOR- * 02/21/89
008500* MINOR CODE IS PLACED INTO A DATABASE FILE AND THE FILE IS * 02/21/89
008600* PRINTED IN HEX USING COPYFILE. * 02/21/89
008700***** 02/21/89
008800* 10/16/87
008900 GETFBA. 10/16/87
74 009000 OPEN OUTPUT DFILE. 10/16/87
75 009100 MOVE MAJ-MIN TO DUMP-MAJ-MIN. 10/16/87
76 009200 MOVE ICFREC TO DUMP-RECORD. 10/16/87
77 009300 WRITE DUMPREC. 10/16/87
78 009400 CLOSE DFILE. 10/16/87

```

Figure 10-9 (Part 2 of 5). Source Program Example — CSDBAT (User-Defined Formats)

```

79 009500 MOVE 31 TO LEN. 10/16/87
80 009600 CALL "QCAEXEC" USING CMD2 LEN. 10/16/87
81 009700 MOVE "PROGRAM TERMINATED DUE TO ERROR IN XPF-ICF FILE" 10/16/87
009800 TO PRINTREC. 10/16/87
82 009900 WRITE PRINTREC. 10/16/87.
83 010000 STOP RUN. 10/16/87
010100* 10/16/87
010200 EXIT-DECLARATIVES. 10/16/87
010300 EXIT. 10/16/87
010400* 02/28/89
84 010500 END DECLARATIVES. 10/16/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDBAT RCHAS400 10/03/90 14:38:26 Page 5
STMT SEQNBR -A 1 B.+...2....+...3....+...4....+...5....+...6....+...7..IDENTFCN S COPYNAME CHG DATE
010600/ 10/16/87
010700 START-PROGRAM SECTION. 10/16/87
010800* 10/16/87
010900 START-PROGRAM-PARAGRAPH. 10/16/87
011000* 3 10/16/87
85 011100 OPEN INPUT DBFILE 10/16/87
011200 I-0 SRCICF 10/16/87
011300 OUTPUT QPRINT. 10/16/87
011400* 10/16/87
011500***** 02/21/89
011600* THE FOLLOWING TEST IS TO ATTEMPT RECOVERY IF AN ERROR OCCURS * 02/21/89
011700* WHEN OPENING THE ICF FILE. * 02/21/89
011800***** 02/21/89
011900* 4 10/16/87
86 012000 IF ERR-SW = "1" 10/16/87
87 012100 THEN IF OPEN-COUNT IS = 9 10/16/87
88 012200 THEN GO TO ABNORMAL-TERMINATION 10/16/87
012300 ELSE 10/16/87
89 012400 ADD 1 TO OPEN-COUNT 10/16/87
90 012500 PERFORM ERROR-RECOVERY-RTN 10/16/87
91 012600 GO TO START-PROGRAM-PARAGRAPH 10/16/87
012700 ELSE 10/16/87
92 012800 MOVE 0 TO OPEN-COUNT. 10/16/87
012900* 10/16/87
013000***** 02/21/89
013100* EVOKE THE PROGRAM "CTDBATCL" ON THE REMOTE SYSTEM IN LIBRARY * 02/21/89
013200* ICFLIB. INDICATOR IN50 IS THE EVOKE KEYWORD. * 02/21/89
013300***** 02/21/89
013400* 5 10/16/87
93 013500 MOVE "CTDBATCL" TO PGMID OF EVOKPGM-0. 10/16/87
94 013600 MOVE "ICFLIB" TO LIB OF EVOKPGM-0. 10/16/87
95 013700 MOVE INDON TO CMNF-INDIC(50). 10/16/87
96 013800 WRITE ICFREC FORMAT IS "EVOKPGM" 10/16/87
013900 INDICATORS ARE CMNF-INDIC-AREA. 10/16/87
97 014000 MOVE INDOFF TO CMNF-INDIC(50) 10/16/87
98 014100 IF ERK-SW = "1" 10/16/87
99 014200 PERFORM ERROR-RECOVERY-RTN 10/16/87
100 014300 GO TO START-PROGRAM-PARAGRAPH. 10/16/87
014400* 10/16/87
014500***** 02/21/89
014600* WHEN THE EVOKE OPERATION IS SUCCESSFUL, A RECORD FROM THE DATA- * 02/21/89
014700* BASE FILE IS READ AND THEN SENT TO THE TARGET SYSTEM. THIS WILL * 02/21/89
014800* BE REPEATED UNTIL THE END OF FILE IS REACHED ON THE DATABASE * 02/21/89
014900* FILE. AT END OF FILE, THE PROGRAM DEVICE IS INVITED AND THE READ * 02/21/89
015000* OPERATION IS ISSUED TO GET THE DATA FROM THE REMOTE SYSTEM. * 02/21/89
015100***** 02/21/89
015200* 02/21/89
015300 SEND-DATA. 10/16/87
015400* 6 10/16/87
101 015500 READ DBFILE 10/16/87
015600 AT END 10/16/87
102 015700 PERFORM INVITE-TO-SEND. 10/16/87
103 015800 IF EOF-DBFILE-SW NOT = "1" 10/16/87
104 015900 MOVE DBREC-DATA TO SNDFLD OF SNDDATA-0 10/16/87
105 016000 PERFORM WRITE-SRCICF-RTN 10/16/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDBAT RCHAS400 10/03/90 14:38:26 Page 6
STMT SEQNBR -A 1 B.+...2....+...3....+...4....+...5....+...6....+...7..IDENTFCN S COPYNAME CHG DATE
106 016100 GO TO SEND-DATA. 10/16/87
016200* 10/16/87
016300***** 02/21/89
016400* A READ OPERATION IS ISSUED TO PROGRAM DEVICE TO CONTINUE * 02/21/89
016500* RECEIVING DATA FROM THE REMOTE SYSTEM UNTIL THE RCVDETACH IND]- * 02/21/89
016600* CATOR IS SET. EACH RECORD RECEIVED WILL BE PRINTED ON THE PRINT * 02/21/89
016700* FILE. * 02/21/89
016800***** 02/21/89
016900* 10/16/87
017000 RECEIVE-DATA. 10/16/87

```

Figure 10-9 (Part 3 of 5). Source Program Example — CSDBAT (User-Defined Formats)

```

017100* 7 10/16/87
107 017200 READ SRCICF INDICATORS ARE CMNF-INDIC-AREA. 10/16/87
108 017300 IF ERR-SW = "1" 10/16/87
109 017400 PERFORM ERROR-RECOVERY-RTN 10/16/87
110 017500 GO TO START-PROGRAM-PARAGRAPH. 10/16/87
111 017600 IF MAJ NOT = "03" 10/16/87
112 017700 MOVE ICFREC TO PRINTREC 10/16/87
113 017800 WRITE PRINTREC. 10/16/87
114 017900 IF CMNF-INDIC(35) NOT = INDON 10/16/87
115 018000 GO TO RECEIVE-DATA. 10/16/87
018100* 10/16/87
018200***** 02/21/89
018300* WHEN PROCESSING IS COMPLETED, END OF JOB MESSAGE IS PRINTED. * 02/21/89
018400* FILES ARE CLOSED AND THE SESSION IS RELEASED. * 02/21/89
018500***** 02/21/89
018600* 10/16/87
018700* 8 10/16/87
116 018800 MOVE "CSDBAT HAS COMPLETED NORMALLY" TO PRINTREC. 10/16/87
117 018900 WRITE PRINTREC. 10/16/87
118 019000 CLOSE DBFILE 10/16/87
019100 SRCICF 10/16/87
019200 QPRINT. 10/16/87
119 019300 STOP RUN. 10/16/87
019400* 10/16/87
019500***** 02/21/89
019600* THIS SUBROUTINE SENDS DATA TO THE REMOTE SYSTEM * 02/21/89
019700***** 02/21/89
019800 WRITE-SRCICF-RTN. 10/16/87
019900* 9 10/16/87
120 020000 WRITE ICFREC FORMAT IS "SNDDATA" 10/16/87
020100 INDICATORS ARE CMNF-INDIC-AREA. 10/16/87
121 020200 IF ERR-SW = "1" 10/16/87
122 020300 PERFORM ERROR-RECOVERY-RTN 10/16/87
123 020400 GO TO START-PROGRAM-PARAGRAPH. 10/16/87
020500* 10/16/87
020600 ERROR-RECOVERY-RTN. 10/16/87
020700* 10 10/16/87
124 020800 CLOSE SRCICF 10/16/87
020900 DBFILE 10/16/87
021000 QPRINT. 10/16/87
125 021100 MOVE "0" TO ERR-SW. 10/16/87
021200* 10/16/87
021300***** 02/21/89
021400* WHEN AN ERROR OCCURS ON AN ICF SESSION, INFORMATION ABOUT THE * 02/21/89
021500* ERROR IS PRINTED. * 02/21/89
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDBAT RCHAS400 10/03/90 14:38:26 Page 7
STMT SEQNBR -A 1 B. ....2.....3.....4.....5.....6.....7..IDENTFCN S COPYNAME CHG DATE
021600***** 02/21/89
021700 ABNORMAL-TERMINATION. 10/16/87
021800* 11 10/16/87
126 021900 MOVE "CSDBAT HAS COMPLETED ABNORMALLY" 10/16/87
022000 TO PRINTREC. 10/16/87
127 022100 WRITE PRINTREC. 10/16/87
128 022200 STOP RUN. 10/16/87
022300* 10/16/87
022400***** 02/21/89
022500* WHEN END OF FILE IS DETECTED, AN INVITE OPERATION IS ISSUED TO * 02/21/89
022600* NOTIFY THE TARGET THAT IT CAN START SENDING DATA. * 02/21/89
022700***** 02/21/89
022800 INVITE-TO-SEND. 10/16/87
022900* 12 10/16/87
129 023000 MOVE INDON TO CMNF-INDIC(45). 10/16/87
130 023100 MOVE "1" TO EOF-DBFILE-SW. 10/16/87
131 023200 WRITE ICFREC FORMAT IS "INVITE" 10/16/87
023300 INDICATORS ARE CMNF-INDIC-AREA. 10/16/87
132 023400 IF ERR-SW = "1" 10/16/87
133 023500 PERFORM ERROR-RECOVERY-RTN 10/16/87
134 023600 GO TO START-PROGRAM-PARAGRAPH. 10/16/87
***** END OF SOURCE *****

```

Figure 10-9 (Part 4 of 5). Source Program Example — CSDBAT (User-Defined Formats)

```

STMT
* 19 MSGID: LBL0650 SEVERITY: 00 SEQNBR: 002700
Message . . . . : Blocking/Deblocking for file 'DBFILE' will
be performed by compiler-generated code.
* 26 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 003400
Message . . . . : No INPUT fields found for format EVOKPGM.
* 26 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 003400
Message . . . . : No INPUT fields found for format ENDREC.
* 26 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 003400
Message . . . . : No OUTPUT fields found for format ENDREC.
* 26 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 003400
Message . . . . : No INPUT fields found for format INVITE.
* 26 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 003400
Message . . . . : No OUTPUT fields found for format INVITE.
* 39 MSGID: LBL0650 SEVERITY: 00 SEQNBR: 003500
Message . . . . : Blocking/Deblocking for file 'DFILE' will be
performed by compiler-generated code.
***** END OF MESSAGES *****
Message Summary
Total      Info(0-4)      Warning(5-19)      Error(20-29)      Severe(30-39)      Terminal(40-99)
7          2          5          0          0          0
Source records read . . . . . : 236
Copy records read . . . . . : 37
Copy members processed . . . . . : 1
Sequence errors . . . . . : 0
Highest severity message issued . . : 10
LBL0901 00 Program CSDBAT created in library ICFLIB.
***** END OF COMPILATION *****

```

Figure 10-9 (Part 5 of 5). Source Program Example — CSDBAT (User-Defined Formats)

```

5738CB1 V2R1M0 910524          IBM AS/400 COBOL/400          ICFLIB/CSFBAT          RCHAS400 10/03/90 14:35:04  Page 1
Program . . . . . : CSFBAT
Library . . . . . : ICFLIB
Source file . . . . . : QICFPUB
Library . . . . . : ICFLIB
Source member . . . . . : CSFBAT 10/03/90 14:27:28
Generation severity level . . . . . : 29
Text 'description' . . . . . : cobol batch file transfer using $$FORMAT (source)
Source listing options . . . . . : *SOURCE
Generation options . . . . . : *NONE
Message limit:
  Number of messages . . . . . : *NOMAX
  Message limit severity . . . . . : 29
Print file . . . . . : QSYSPRT
Library . . . . . : *LIBL
FIPS flagging . . . . . : *NOFIPS *NOSEG *NODEB *NOBSOLETE
SAA flagging . . . . . : *NOFLAG
Flagging severity . . . . . : 0
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Compiler . . . . . : IBM AS/400 COBOL/400
5738CB1 V2R1M0 910524          AS/400 COBOL Source          ICFLIB/CSFBAT          RCHAS400 10/03/90 14:35:04  Page 2

```

```

STMT SEQNBR -A 1 B...2...3...4...5...6...7..IDENTFCN S COPYNAME CHG DATE
1 000100 IDENTIFICATION DIVISION.                                02/21/89
2 000200 PROGRAM-ID. CSFBAT.                                     02/21/89
000300*****
000400* THIS IS A BATCH FILE TRANSFER PROGRAM THAT READS A SEQUENTIAL * 02/21/89
000500* FILE AND SENDS THE RECORDS TO THE REMOTE SYSTEM UNTIL THE END * 02/21/89
000600* OF FILE IS REACHED. AT THIS TIME, THE PROGRAM STOPS SENDING * 02/21/89
000700* AND STARTS RECEIVING RECORDS FROM THE REMOTE SYSTEM UNTIL * 02/21/89
000800* A DETACH INDICATION IS RECEIVED. * 02/21/89
000900*****
3 001000 ENVIRONMENT DIVISION.                                    10/16/87
4 001100 CONFIGURATION SECTION.                                  10/16/87
5 001200 SOURCE-COMPUTER. IBM-AS400.                             02/21/89
6 001300 OBJECT-COMPUTER. IBM-AS400.                             02/21/89
7 001400 SPECIAL-NAMES. I-O-FEEDBACK IS FEEDBACK-AREA          10/16/87
8 001500 OPEN-FEEDBACK IS OPEN-FBA.                             10/16/87
9 001600 INPUT-OUTPUT SECTION.                                   10/16/87
10 001700 FILE-CONTROL.                                         10/16/87
001800* 1
11 001900 SELECT DBFILE ASSIGN TO DATABASE-DBFILE.                10/16/87
12 002000 SELECT SRCICF ASSIGN TO WORKSTATION-SRCICF-SI          10/16/87
13 002100 ORGANIZATION IS TRANSACTION                            10/16/87
14 002200 FILE STATUS IS STATUS-IND MAJ-MIN.                    10/16/87
15 002300 SELECT DFILE ASSIGN TO DATABASE-HEXDUMP.               10/16/87
16 002400 SELECT QPRINT ASSIGN TO PRINTER-QSYSPRT.              10/16/87
17 002500 DATA DIVISION.                                       10/16/87
18 002600 FILE SECTION.                                         10/16/87
19 002700 FD DBFILE                                             10/16/87
20 002800 LABEL RECORDS ARE STANDARD.                            10/16/87
21 002900 01 DBREC.                                             10/16/87
22 003000 05 DBREC-DATA PIC X(80).                               10/16/87
23 003100 FD SRCICF                                             10/16/87
24 003200 LABEL RECORDS ARE STANDARD.                            10/16/87
25 003300 01 ICFREC.                                           10/16/87
26 003400 05 EVOKPGM-0.                                         10/16/87
27 003500 10 PGMID PIC X(8).                                     10/16/87
28 003600 10 PASSWD PIC X(8).                                    11/16/88
29 003700 10 USERID PIC X(8).                                    11/16/88
30 003800 10 LIB PIC X(8).                                       10/16/87
31 003900 10 FILLER PIC X(52).                                   10/16/87
32 004000 05 SNDDATA-0 REDEFINES EVOKPGM-0.                    10/16/87
33 004100 10 SNDLENGTH PIC 9(4).                                  10/16/87
34 004200 10 SNDFIELD PIC X(80).                                  10/16/87
35 004300 05 INVITE-0 REDEFINES EVOKPGM-0.                      02/27/89
36 004400 10 INVLENGTH PIC 9(4).                                  10/16/87
37 004500 10 INVFIELD PIC X(80).                                 10/16/87
38 004600 FD DFILE                                             10/16/87
39 004700 LABEL RECORDS ARE STANDARD.                            10/16/87
40 004800 01 DUMPREC.                                           10/16/87

```

Figure 10-10 (Part 1 of 4). Source Program Example — CSFBAT (System-Supplied Formats)

```

41 004900 05 DUMP-MAJ-MIN PIC X(4). 10/16/87
42 005000 05 DUMP-RECORD PIC X(400). 10/16/87
43 005100 FD QPRINT 10/16/87
44 005200 LABEL RECORDS ARE OMITTED. 10/16/87
45 005300 01 PRINTREC PIC X(132). 10/16/87
46 005400 WORKING-STORAGE SECTION. 10/16/87
47 005500 77 STATUS-IND PIC X(2). 10/16/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICLLIB/CSFBAT RCHAS400 10/03/90 14:35:04 Page 3
STMT SEQNBR -A 1 B. ....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
48 005600 77 MAJ-MIN-SAV PIC X(4). 10/16/87
49 005700 77 EOF-DBFILE-SW PIC X VALUE "0". 10/16/87
50 005800 77 ERR-SW PIC X VALUE "0". 10/16/87
51 005900 77 OPEN-COUNT PIC 9(1) VALUE 0. 10/16/87
52 006000 77 LEN PIC 9(10)V9(5) COMP. 10/16/87
006100 10/16/87
53 006200 77 CMD2 PIC X(31) 10/16/87
54 006300 VALUE "CPYF HEXDUMP *LIST PRTFMT(*HEX)". 10/16/87
55 006400 01 OPEN-FBA. 10/16/87
56 006500 05 FILLER PIC X(75). 10/16/87
57 006600 05 RECS-IN-DB PIC 9(09) COMP-4. 10/16/87
58 006700 05 FILLER PIC X(45). 10/16/87
59 006800 01 MAJ-MIN. 10/16/87
60 006900 05 MAJ PIC X(2). 10/16/87
61 007000 05 MIN PIC X(2). 10/16/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICLLIB/CSFBAT RCHAS400 10/03/90 14:35:04 Page 4
STMT SEQNBR -A 1 B. ....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
007100/ 10/16/87
62 007200 PROCEDURE DIVISION. 10/16/87
007300 DECLARATIVES. 10/16/87
007400 ERR-SECTION SECTION. 10/16/87
007500* 2 10/16/87
007600 USE AFTER STANDARD ERROR PROCEDURE ON SRCICF. 10/16/87
007700 SRCICF-EXCEPTION. 10/16/87
007800* 10/16/87
007900* CHECK THE MAJOR/MINOR CODES AND TAKE APPROPRIATE ACTION 10/16/87
008000* 10/16/87
008100* RECOVERABLE SESSION ERROR. CLOSE ICF FILE. 10/03/90
63 008200 IF MAJ = "03" OR MAJ = "83" 10/16/87
64 008300 MOVE "PROGRAM STARTED AGAIN DUE TO SESSION ERROR" 10/16/87
008400 TO PRINTREC 10/16/87
65 008500 WRITE PRINTREC 10/16/87
66 008600 MOVE "1" TO ERR-SW 10/16/87
67 008700 GO TO EXIT-DECLARATIVES. 10/16/87
008800* 10/16/87
008900***** 02/21/89
009000* WHEN THERE IS A PERMANENT SESSION ERROR DETECTED, THE MAJOR-MINOR* 02/21/89
009100* CODE IS PLACED INTO A DATABASE FILE AND THE FILE IS PRINTED IN * 02/21/89
009200* HEX USING COPYFILE. * 02/21/89
009300***** 02/21/89
009400* 10/16/87
009500 GETFBA. 10/16/87
68 009600 OPEN OUTPUT DFILE. 10/16/87
69 009700 MOVE MAJ-MIN TO DUMP-MAJ-MIN. 10/16/87
70 009800 MOVE ICFREC TO DUMP-RECORD. 10/16/87
71 009900 WRITE DUMPREC. 10/16/87
72 010000 CLOSE DFILE. 10/16/87
73 010100 MOVE 31 TO LEN. 10/16/87
74 010200 CALL "QCAEXEC" USING CMD2 LEN. 10/16/87
75 010300 MOVE "PROGRAM TERMINATED DUE TO ERROR IN ICF FILE" 10/03/90
010400 TO PRINTREC. 10/16/87
76 010500 WRITE PRINTREC. 10/16/87
77 010600 STOP RUN. 10/16/87
010700* 10/16/87
010800 EXIT-DECLARATIVES. 10/16/87
010900 EXIT. 02/28/89
011000* 10/16/87
78 011100 END DECLARATIVES. 10/16/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICLLIB/CSFBAT RCHAS400 10/03/90 14:35:04 Page 5
STMT SEQNBR -A 1 B. ....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
011200/ 10/16/87
011300 START-PROGRAM SECTION. 10/16/87
011400* 10/16/87
011500 START-PROGRAM-PARAGRAPH. 10/16/87
011600* 3 10/16/87
79 011700 OPEN INPUT DBFILE 10/16/87
011800 1-0 SRCICF 10/16/87
011900 OUTPUT QPRINT. 10/16/87
012000* 10/16/87

```

Figure 10-10 (Part 2 of 4). Source Program Example — CSFBAT (System-Supplied Formats)

```

012100*****
012200* THE FOLLOWING TEST IS TO ATTEMPT RECOVERY IF AN ERROR OCCURS *
012300* WHEN OPENING THE ICF FILE. *
012400*****
012500* 4
80 012600 IF ERR-SW = "1"
81 012700 THEN IF OPEN-COUNT IS = 9
82 012800 THEN GO TO ABNORMAL-TERMINATION
012900 ELSE
83 013000 ADD 1 TO OPEN-COUNT
84 013100 PERFORM ERROR-RECOVERY-RTN
85 013200 GO TO START-PROGRAM-PARAGRAPH
013300 ELSE
86 013400 MOVE 0 TO OPEN-COUNT.
013500*
013600*****
013700* EVOKE THE PROGRAM "CTDBATCL" ON THE REMOTE SYSTEM IN LIBRARY *
013800* ICFLIB. INDICATOR IN50 IS THE EVOKE KEYWORD. *
013900*****
014000* 5
87 014100 MOVE SPACES TO EVOKPGM-0.
88 014200 MOVE "CTFBATCL" TO PGMID OF EVOKPGM-0.
89 014300 MOVE "QSECOFR" TO PASSWD OF EVOKPGM-0.
90 014400 MOVE "QSECOFR" TO USERID OF EVOKPGM-0.
91 014500 MOVE "ICFLIB" TO LIB OF EVOKPGM-0.
92 014600 WRITE ICFREC FORMAT IS "$$EVOKNI".
93 014700 IF ERR-SW = "1"
94 014800 PERFORM ERROR-RECOVERY-RTN
95 014900 GO TO START-PROGRAM-PARAGRAPH.
015000*
015100*****
015200* WHEN THE EVOKE OPERATION IS SUCCESSFUL, A RECORD FROM THE DATA- *
015300* BASE FILE IS READ AND THEN SENT TO THE TARGET SYSTEM. THIS WILL *
015400* BE REPEATED UNTIL THE END OF FILE IS REACHED ON THE DATABASE *
015500* FILE. AT END OF FILE, THE PROGRAM DEVICE IS INVITED AND THE READ *
015600* OPERATION IS ISSUED TO GET THE DATA FROM THE REMOTE SYSTEM. *
015700*****
015800*
015900 SEND-DATA.
016000* 6
96 016100 READ DBFILE
016200 AT END
97 016300 PERFORM INVITE-TO-SEND.
98 016400 IF EOF-DBFILE-SW NOT = "1"
99 016500 MOVE DBREC-DATA TO SNDFIELD OF SNDDATA-0
100 016600 MOVE +80 TO SNDLENGTH OF SNDDATA-0
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFBAT RCHAS400 10/03/90 14:35:04 Page 6
STMT SEONBR -A 1 B...+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
101 016700 PERFORM WRITE-SRCLCF-RTN
102 016800 GO TO SEND-DATA.
016900*
017000*****
017100* A READ OPERATION IS ISSUED TO PROGRAM DEVICE TO CONTINUE *
017200* RECEIVING DATA FROM THE REMOTE SYSTEM UNTIL THE RCVDDETACH *
017300* INDICATOR IS SET. EACH RECORD RECEIVED WILL BE PRINTED TO THE *
017400* PRINT FILE. *
017500*****
017600*
017700 RECEIVE-DATA.
017800* 7
103 017900 READ SRCLCF.
104 018000 IF ERR-SW = "1"
105 018100 PERFORM ERROR-RECOVERY-RTN
106 018200 GO TO START-PROGRAM-PARAGRAPH.
107 018300 IF MAJ NOT = "03"
108 018400 MOVE ICFREC TO PRINTREC
109 018500 WRITE PRINTREC.
110 018600 IF MIN NOT = "08"
111 018700 GO TO RECEIVE-DATA.
018800*

```

Figure 10-10 (Part 3 of 4). Source Program Example — CSFBAT (System-Supplied Formats)

```

018900*****
019000* WHEN PROCESSING IS COMPLETED, END OF JOB MESSAGE IS PRINTED. *
019100* FILES ARE CLOSED AND THE SESSION IS ENDED. *
019200*****
019300*
019400* 8
112 019500 MOVE "CSFBAT HAS COMPLETED NORMALLY" TO PRINTREC.
113 019600 WRITE PRINTREC.
114 019700 CLOSE DBFILE
019800 SRCICF
019900 QPRINT.
115 020000 STOP RUN.
020100*
020200*****
020300* THIS SUBROUTINE SENDS DATA TO THE REMOTE SYSTEM *
020400*****
020500 WRITE-SRCICF-RTN.
020600* 9
116 020700 WRITE ICFREC FORMAT IS "$$SENDNI".
117 020800 IF ERR-SW = "1"
118 020900 PERFORM ERROR-RECOVERY-RTN
119 021000 GO TO START-PROGRAM-PARAGRAPH.
021100*
021200 ERROR-RECOVERY-RTN.
021300* 10
120 021400 CLOSE SRCICF
021500 DBFILE
021600 QPRINT.
121 021700 MOVE "0" TO ERR-SW.
021800*
021900*****
022000* WHEN AN ERROR OCCURS ON AN ICF SESSION, INFORMATION ABOUT *
022100* THE ERROR IS PRINTED. *
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFBAT RCHAS400 10/03/90 14:35:04 Page 7
STMT SEQNBR -A 1 B.+. . . . 2. . . . +. . . . 3. . . . +. . . . 4. . . . +. . . . 5. . . . +. . . . 6. . . . +. . . . 7. . IDENTFCN S COPYNAME CHG DATE
022200*****
022300 ABNORMAL-TERMINATION.
022400* 11
122 022500 MOVE "CSFBAT HAS COMPLETED ABNORMALLY"
022600 TO PRINTREC.
123 022700 WRITE PRINTREC.
124 022800 STOP RUN.
022900*
023000*****
023100* WHEN END OF FILE IS DETECTED, AN INVITE OPERATION IS ISSUED TO *
023200* NOTIFY THE TARGET THAT IT CAN START SENDING DATA. *
023300*****
023400 INVITE-TO-SEND.
023500* 12
125 023600 MOVE "1" TO EOF-DBFILE-SW.
126 023700 MOVE +0 TO INVLENGTH OF INVITE-0.
127 023800 WRITE ICFREC FORMAT IS "$$SEND".
128 023900 IF ERR-SW = "1"
129 024000 PERFORM ERROR-RECOVERY-RTN
130 024100 GO TO START-PROGRAM-PARAGRAPH.
***** END OF SOURCE *****
5738CB1 V2R1M0 910524 AS/400 COBOL Messages ICFLIB/CSFBAT RCHAS400 10/03/90 14:35:04 Page 8
STMT
* 19 MSGID: LBL0650 SEVERITY: 00 SEQNBR: 002700
Message . . . . : Blocking/Deblocking for file 'DBFILE' will
be performed by compiler-generated code.
* 38 MSGID: LBL0650 SEVERITY: 00 SEQNBR: 004600
Message . . . . : Blocking/Deblocking for file 'DFILE' will be
performed by compiler-generated code.
***** END OF MESSAGES *****
Message Summary
Total Info(0-4) Warning(5-19) Error(20-29) Severe(30-39) Terminal(40-99)
2 2 0 0 0 0
Source records read . . . . . : 241
Copy records read . . . . . : 0
Copy members processed . . . . . : 0
Sequence errors . . . . . : 0
Highest severity message issued . . : 0
LBL0901 00 Program CSFBAT created in library ICFLIB.
***** END OF COMPILATION *****

```

Figure 10-10 (Part 4 of 4). Source Program Example — CSFBAT (System-Supplied Formats)

Target Program Batch Transfer

(Example I): The following describes a COBOL target program batch transfer program.

Program Files: The COBOL batch transfer target program uses the following files:

TGTICF

An ICF file used to send records to and receive records from the source program.

DBFILE

A database file that contains the records to be sent to the source program.

QPRINT

A printer file used to print the records received from the source program.

DDS Source: The DDS source used in the ICF file is illustrated in the following example. The other files (DBFILE and QPRINT) are program-described and therefore requires no DDS.

```
A*****
A*                                     *
A*          ICF FILE                   *
A*      USED IN BATCH DATA TRANSFER PROGRAM *
A*                                     *
A*****
A* FILE LEVEL INDICATORS:
A*
A*          INDARA
A*          RCVTRNRND(15 'END OF DATA')
A*
A 38          DETACH
A*
A*          INDXTX(30 '38->DETACH TARG-
A*          GET PROGRAM.')
```

```
A*          RCVDATCH(35 'RECEIVED
A*          DETACHED.')
```

```
A*****
A*          ICF RECORD FORMATS        *
A*****
A*          R RCVDATA
A*          RCVFLO          88A
A*          R SNDDATA
A*          SNDFLO          88A
A*          R EVOKPGM
A 50          EVOKE(&LIB/&PGMID)
A 50          SECURITY(2 'PASSWRD' +
A*          3 'USERID')
```

```
A          PGMID          18A P
A          ITR            18A P
```

ICF File Creation and Program Device Entry

Definition: The command needed to create the ICF file is:

```
CRTICFF FILE(ICFLIB/TGTICF) SRCFILE(ICFLIB/QICFPUB)
SRCMBR(TGTICF) ACQPGMDEV(PGMDEVB) TEXT('TARGET ICF
FILE FOR BATCH DATA TRANSFER')
```

The command needed to define the program device entry is:

```
ADDICFDEVE FILE(ICFLIB/TGTICF) PGMDEV(PGMDEVB)
RMTLOCNAME(*REQUESTER)
```

This example acquires all sessions at the beginning of the program. For performance considerations, you may not want to acquire sessions until they are actually needed in the program.

Program Explanation: The following describes the structure of the program examples illustrated in Figure 10-11 on page 10-19 and Figure 10-12 on page 10-23. The ICF file used in the first example is defined by the user, and uses externally described data formats (DDS). The second example uses the same file, but uses program-described data and system-supplied formats. The reference numbers in the explanation below correspond to the numbers in the following program examples.

Although the basic structure of the two examples provided is the same, there are differences because of the use of user-defined formats and system-supplied formats.

Differences between the first and second example are documented as notes in each of the descriptions.

- 1 This section identifies the files used in the program. TGTICF is the ICF file used to send records to the source program.

This section also contains declarations of I/O variables, work areas, and constants needed. MAJ-MIN contains the major/minor return code from the I/O feedback area.

Note: In the program using system-supplied formats, the input records for TGTICF are explicitly coded in the program since TGTICF is now treated as a program-described file. The system-supplied file, QICDMF, can be used instead of TGTICF. Using the system-supplied file can be done by specifying QICDMF in the file specification, or by using an OVRICFF command to change the file name from TGTICF to QICDMF. The OVRICFF command can also be used to change the ACQPGMDEV parameter of the file.

- 2 The program defines the error handling for the program. If an error occurs, the program writes the following message to the print file:

```
CTDBAT HAS COMPLETED ABNORMALLY
```

The session is released.

Note: In the program using system-supplied formats, the program name is CTFBAT.

- 3** The program opens the files that are going to be used as follows:

DBFILE

The database file used as input for transmitting to the source program

QPRINT

The printer file used as output for records received

TGTICF

The ICF file used to receive data from and send data to the source program

Because the ICF file was created using the ACQPGMDEV parameter, the program device is automatically acquired when the file is opened. Therefore, no acquire operation is coded in the program.

- 4** Data is read from the program device (TGTICF file).

If an error occurs on the read (major return code greater than '03'), control passes to **2**. Otherwise if data was received (major return code not = '03'), then the data is written to the printer file (QPRINT).

- 5** Data records are read until the change-direction indicator is received from the source program. When change direction is received, indicator 15 is set on, as defined by the RCVTRNRND keyword in the DDS for the ICF file, and control is passed to **6**.

Note: In the program using system-supplied formats, the minor return code of '00' is checked to verify whether change direction is received.

- 6** The database file is read and the records sent to the source program until the end of the database file. At end-of-file, the program passes control to **7**.

If it is not the last record, then the data is written to the program device using the format SNDDATA, and the next database record is read. If an error occurs on the write operation, the program goes to **2** and a message is printed.

Note: In the program using system-supplied formats, the \$\$\$SENDNI format is used instead of the user-defined SNDDATA format.

- 7** This routine issues a detach request to the program device using format ENDREC. Indicator 30 is associated with the DETACH keyword. If an error occurs, the program goes to **2** and a message is printed.

Note: In the program using system-supplied formats, the \$\$\$SENDET format is used instead of the user-defined ENDREC format.

- 8** After the detach request has been sent, the following message is written to the print file:

```
CTDBAT HAS COMPLETED NORMALLY
```

The files are closed. The program device is automatically released as a result of the close operation, and the program ends.

Note: In the program using system-supplied formats, the program name is CTFBAT.

```

5738CB1 V2R1M0 910524          IBM AS/400 COBOL/400          ICFLIB/CTDBAT          RCHAS400 10/03/90 14:38:53  Page 1
Program . . . . . : CTDBAT
Library . . . . . : ICFLIB
Source file . . . . . : QICFPUB
Library . . . . . : ICFLIB
Source member . . . . . : CTDBAT 02/21/89 17:51:20
Generation severity level . . . . . : 29
Text 'description' . . . . . : cobol batch file transfer using dds (target)
Source listing options . . . . . : *SOURCE
Generation options . . . . . : *NONE
Message limit:
  Number of messages . . . . . : *NOMAX
  Message limit severity . . . . . : 29
Print file . . . . . : QSYSPT
Library . . . . . : *LIBL
FIPS flagging . . . . . : *NOFIPS *NOSEJ *NODEB *NOOBSOLETE
SAA flagging . . . . . : *NOFLAG
Flagging severity . . . . . : 0
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Compiler . . . . . : IBM AS/400 COBOL/400
5738CB1 V2R1M0 910524          AS/400 COBOL Source          ICFLIB/CTDBAT          RCHAS400 10/03/90 14:38:53  Page 2

```

```

STMT SEQNBR -A 1 B.+. . . . . 2. . . . . 3. . . . . 4. . . . . 5. . . . . 6. . . . . 7. . . . . IDENTFCN S COPYNAME CHG DATE
1 000100 IDENTIFICATION DIVISION.                                02/21/89
2 000200 PROGRAM-ID. CTDBAT.                                     02/21/89
000300*****
000400* THIS TARGET PROGRAM IS EVOKED BY THE SOURCE PROGRAM AND * 02/21/89
000500* RECEIVES RECORDS FROM IT. WHEN THE SOURCE PROGRAM IS DONE * 02/21/89
000600* SENDING DATA, THIS PROGRAM SENDS ITS OWN RECORDS UNTIL IT IS * 02/21/89
000700* DONE. WHEN IT IS DONE, IT SENDS A DETACH REQUEST TO THE SOURCE* 02/21/89
000800* PROGRAM AND ENDS ITS SESSION AND JOB. * 02/21/89
000900*****
3 001000 ENVIRONMENT DIVISION.                                    10/16/87
4 001100 CONFIGURATION SECTION.                                  10/16/87
5 001200 SOURCE-COMPUTER. IBM-AS400.                             02/21/89
6 001300 OBJECT-COMPUTER. IBM-AS400.                             02/21/89
7 001400 SPECIAL-NAMES. I-O-FEEDBACK IS FEEDBACK-AREA           10/16/87
8 001500 OPEN-FEEDBACK IS OPEN-FBA.                               10/16/87
9 001600 INPUT-OUTPUT SECTION.                                    10/16/87
001700* 1
10 001800 FILE-CONTROL.                                          10/16/87
11 001900 SELECT DBFILE ASSIGN TO DATABASE-DBFILE.              10/16/87
12 002000 SELECT TGTICF ASSIGN TO WORKSTATION-TGTICF-SI         10/16/87
13 002100 ORGANIZATION IS TRANSACTION                            10/16/87
14 002200 FILE STATUS IS STATUS-IND MAJ-MIN.                     10/16/87
15 002300 SELECT DFILE ASSIGN TO DATABASE-HEXDUMP.              10/16/87
16 002400 SELECT QPRINT ASSIGN TO PRINTER-QSYSPT.               10/16/87
17 002500 DATA DIVISION.                                        10/16/87
18 002600 FILE SECTION.                                         10/16/87
19 002700 FD DBFILE                                             10/16/87
20 002800 LABEL RECORDS ARE STANDARD.                            10/16/87
21 002900 01 DBREC.                                             10/16/87
22 003000 05 DBREC-DATA PIC X(80).                               10/16/87
23 003100 FD TGTICF                                             10/16/87
24 003200 LABEL RECORDS ARE STANDARD.                            10/16/87
25 003300 01 ICFREC.                                             10/16/87
26 003400 COPY DDS-ALL-FORMATS OF TGTICF.                       10/16/87
27 +000001 05 TGTICF-RECORD PIC X(80).                           <-ALL-FMTS
+000002* INPUT FORMAT:RCVDATA FROM FILE TGTICF OF LIBRARY ICFLIB <-ALL-FMTS
+000003* <-ALL-FMTS
28 +000004 05 RCVDATA-I REDEFINES TGTICF-RECORD.                 <-ALL-FMTS
29 +000005 06 RCVFLD PIC X(80). <-ALL-FMTS
+000006* OUTPUT FORMAT:RCVDATA FROM FILE TGTICF OF LIBRARY ICFLIB <-ALL-FMTS
+000007* <-ALL-FMTS
30 +000008 05 RCVDATA-0 REDEFINES TGTICF-RECORD.                 <-ALL-FMTS
31 +000009 06 RCVFLD PIC X(80). <-ALL-FMTS
+000010* INPUT FORMAT:SNDDATA FROM FILE TGTICF OF LIBRARY ICFLIB <-ALL-FMTS
+000011* <-ALL-FMTS
32 +000012 05 SNDDATA-I REDEFINES TGTICF-RECORD.                 <-ALL-FMTS
33 +000013 06 SNDFLD PIC X(80). <-ALL-FMTS
+000014* OUTPUT FORMAT:SNDDATA FROM FILE TGTICF OF LIBRARY ICFLIB <-ALL-FMTS
+000015* <-ALL-FMTS
34 +000016 05 SNDDATA-0 REDEFINES TGTICF-RECORD.                 <-ALL-FMTS
35 +000017 06 SNDFLD PIC X(80). <-ALL-FMTS
+000018* INPUT FORMAT:EVOKPGM FROM FILE TGTICF OF LIBRARY ICFLIB <-ALL-FMTS
+000019* <-ALL-FMTS
+000020* 05 EVOKPGM-I REDEFINES TGTICF-RECORD.                 <-ALL-FMTS
+000021* OUTPUT FORMAT:EVOKPGM FROM FILE TGTICF OF LIBRARY ICFLIB <-ALL-FMTS

```

Figure 10-11 (Part 1 of 4). Target Program Example — CTDBAT (User-Defined Formats)

```

5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTDBAT RCHAS400 10/03/90 14:38:53 Page 3
STMT SEQNBR -A 1 B...2...3...4...5...6...7..IDENTFCN S COPYNAME CHG DATE
+000022* <-ALL-FMTS
36 +000023 05 EVOKPGM-0 REDEFINES TGTCF-RECORD. <-ALL-FMTS
37 +000024 06 PGMID PIC X(10). <-ALL-FMTS
38 +000025 06 LIB PIC X(10). <-ALL-FMTS
+000026* INPUT FORMAT:ENDREC FROM FILE TGTCF OF LIBRARY ICFLIB <-ALL-FMTS
+000027* <-ALL-FMTS
+000028* 05 ENDREC-1 REDEFINES TGTCF-RECORD. <-ALL-FMTS
+000029* OUTPUT FORMAT:ENDREC FROM FILE TGTCF OF LIBRARY ICFLIB <-ALL-FMTS
+000030* <-ALL-FMTS
+000031* 05 ENDREC-0 REDEFINES TGTCF-RECORD. <-ALL-FMTS
+000032* INPUT FORMAT:INVITE FROM FILE TGTCF OF LIBRARY ICFLIB <-ALL-FMTS
+000033* <-ALL-FMTS
+000034* 05 INVITE-1 REDEFINES TGTCF-RECORD. <-ALL-FMTS
+000035* OUTPUT FORMAT:INVITE FROM FILE TGTCF OF LIBRARY ICFLIB <-ALL-FMTS
+000036* <-ALL-FMTS
+000037* 05 INVITE-0 REDEFINES TGTCF-RECORD. <-ALL-FMTS
39 003500 FD DFILE 10/16/87
40 003600 LABEL RECORDS ARE STANDARD. 10/16/87
41 003700 01 DUMPREC. 10/16/87
42 003800 05 DUMP-MAJ-MIN PIC X(4). 10/16/87
43 003900 05 DUMP-RECORD PIC X(80). 10/16/87
44 004000 FD QPRINT 10/16/87
45 004100 LABEL RECORDS ARE OMITTED. 10/16/87
46 004200 01 PRINTREC PIC X(132). 10/16/87
47 004300 WORKING-STORAGE SECTION. 10/16/87
48 004400 77 MAJ-MIN-SAV PIC X(4). 10/16/87
49 004500 77 STATUS-IND PIC X(2). 10/16/87
50 004600 77 INDOON PIC 1 VALUE B"1". 10/16/87
51 004700 77 INDOFF PIC 1 VALUE B"0". 10/16/87
52 004800 77 LEN PIC 9(10)V9(5) COMP 10/16/87
53 004900 VALUE 0. 10/16/87
54 005000 77 CMD2 PIC X(31) 10/16/87
55 005100 VALUE "CPYF HEXDUMP *LIST PRTFMT(*HEX)". 10/16/87
56 005200 01 CMNF-INDIC-AREA. 10/16/87
57 005300 05 CMNF-INDIC PIC 1 OCCURS 99 TIMES 10/16/87
58 005400 INDICATOR 1. 10/16/87
59 005500 01 OPEN-FBA. 10/16/87
60 005600 05 FILLER PIC X(75). 10/16/87
61 005700 05 RECS-IN-DB PIC 9(09) COMP-4. 10/16/87
62 005800 05 FILLER PIC X(45). 10/16/87
63 005900 01 MAJ-MIN. 10/16/87
64 006000 05 MAJ PIC X(2). 10/16/87
65 006100 05 MIN PIC X(2). 10/16/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTDBAT RCHAS400 10/03/90 14:38:53 Page 4
STMT SEQNBR -A 1 B...2...3...4...5...6...7..IDENTFCN S COPYNAME CHG DATE
006200/ 10/16/87
66 006300 PROCEDURE DIVISION. 10/16/87
006400 DECLARATIVES. 10/16/87
006500 ERR-SECTION SECTION. 10/16/87
006600* 2 10/16/87
006700* 10/16/87
006800 USE AFTER STANDARD ERROR PROCEDURE ON TGTCF. 10/16/87
006900 TGTCF-EXCEPTION. 10/16/87
007000* 10/16/87
007100***** 02/21/89
007200* GET INFORMATION FROM THE MAJOR-MINOR CODE AND PLACE IT INTO * 02/21/89
007300* A DATABASE FILE. THEN PRINT THE FILE IN HEX USING COPYFILE. * 02/21/89
007400***** 02/21/89
007500* 10/16/87
007600 GETFBA. 10/16/87
67 007700 MOVE "CTDBAT HAS COMPLETED ABNORMALLY" TO PRINTREC. 10/16/87
68 007800 WRITE PRINTREC. 10/16/87
69 007900 OPEN OUTPUT DFILE. 10/16/87
70 008000 MOVE MAJ-MIN TO DUMP-MAJ-MIN. 10/16/87
71 008100 MOVE ICFREC TO DUMP-RECORD. 10/16/87
72 008200 WRITE DUMPREC. 10/16/87
73 008300 CLOSE DFILE. 10/16/87
74 008400 MOVE 31 TO LEN. 10/16/87
75 008500 CALL "QCAEXEC" USING CMD2 LEN. 10/16/87
76 008600 STOP RUN. 10/16/87
008700* 10/16/87
008800 EXIT-DECLARATIVES. 10/16/87
008900 EXIT. 10/16/87
009000* 10/16/87
77 009100 END DECLARATIVES. 10/16/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTDBAT RCHAS400 10/03/90 14:38:53 Page 5
STMT SEQNBR -A 1 B...2...3...4...5...6...7..IDENTFCN S COPYNAME CHG DATE
009200/ 10/16/87

```

Figure 10-11 (Part 2 of 4). Target Program Example — CTDBAT (User-Defined Formats)

```

009300 START-PROGRAM SECTION.
009400 START-PROGRAM-PARAGRAPH.
009500* 3
78 009600 OPEN OUTPUT QPRINT
009700 1-0 TGTICF
009800 INPUT DBFILE.
009900*
010000*****
010100* DATA CONTINUES TO BE RECEIVED FROM THE PROGRAM DEVICE UNTIL THE *
010200* RCVTRNRND INDICATOR IS SET. EACH RECORD RECEIVED IS PRINTED TO *
010300* THE PRINT FILE. *
010400*****
010500*
010600 RECEIVE-DATA.
010700* 4
79 010800 READ TGTICF INDICATORS ARE CMNF-INDIC-AREA.
80 010900 IF MAJ NOT = "03"
81 011000 MOVE ICFREC TO PRINTREC
82 011100 WRITE PRINTREC.
011200* 5
83 011300 IF CMNF-INDIC(15) NOT = INDON
84 011400 GO TO RECEIVE-DATA.
011500*
011600*****
011700* RECORD IS READ FROM THE DATABASE FILE AND SENT TO THE SOURCE *
011800* PROGRAM. DATA TRANSMISSION CONTINUES UNTIL END OF FILE IS *
011900* DETECTED ON THE DATABASE FILE. AT THIS TIME, A DETACH REQUEST *
012000* IS SENT TO THE SOURCE PROGRAM. *
012100*****
012200*
012300 SEND-DATA.
012400* 6
85 012500 READ DBFILE AT END GO TO SIGNAL-DETACH.
87 012600 WRITE ICFREC FROM DBREC FORMAT IS "SNDDATA"
012700 INDICATORS ARE CMNF-INDIC-AREA.
88 012800 GO TO SEND-DATA.
012900*
013000*****
013100* SIGNAL DETACH TO THE SOURCE PROGRAM. *
013200*****
013300*
013400 SIGNAL-DETACH.
013500* 7
89 013600 MOVE INDON TO CMNF-INDIC(30).
90 013700 WRITE ICFREC FORMAT IS "ENDREC"
013800 INDICATORS ARE CMNF-INDIC-AREA.
013900*****
014000* WHEN THE END OF FILE IS REACHED, AN EOJ MESSAGE IS PRINTED *
014100* AND THE PROGRAM ENDS. *
014200*****
014300* 8
91 014400 MOVE "CTDBAT HAS COMPLETED NORMALLY" TO PRINTREC
92 014500 WRITE PRINTREC.
93 014600 CLOSE DBFILE
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTDBAT RCHAS400 10/03/90 14:38:53 Page 6
STMT SEQNBR -A 1 B.+. . . . 2. . . . +. . . . 3. . . . +. . . . 4. . . . +. . . . 5. . . . +. . . . 6. . . . +. . . . 7. . IDENTFCN S COPYNAME CHG DATE
014700 TGTICF 10/16/87
014800 QPRINT. 10/16/87
94 014900 STOP RUN. 10/16/87
* * * * * E N D O F S O U R C E * * * * *

```

Figure 10-11 (Part 3 of 4). Target Program Example — CTDBAT (User-Defined Formats)

```

5738CB1 V2R1M0 910524          AS/400 COBOL Messages          ICFLIB/CTDBAT          RCHAS400 10/03/90 14:38:53    Page 7
STMT
* 19 MSGID: LBL0650 SEVERITY: 00 SEQNBR: 002700
    Message . . . . : Blocking/Deblocking for file 'DBFILE' will
    be performed by compiler-generated code.
* 26 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 003400
    Message . . . . : No INPUT fields found for format EVOKPGM.
* 26 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 003400
    Message . . . . : No INPUT fields found for format ENDREC.
* 26 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 003400
    Message . . . . : No OUTPUT fields found for format ENDREC.
* 26 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 003400
    Message . . . . : No INPUT fields found for format INVITE.
* 26 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 003400
    Message . . . . : No OUTPUT fields found for format INVITE.
* 39 MSGID: LBL0650 SEVERITY: 00 SEQNBR: 003500
    Message . . . . : Blocking/Deblocking for file 'DFILE' will be
    performed by compiler-generated code.
* 67 MSGID: LBL0335 SEVERITY: 00 SEQNBR: 007600
    Message . . . . : Empty paragraph or section precedes 'GETFBA'
    paragraph or section.
    * * * * * E N D   O F   M E S S A G E S   * * * * *
                                Message Summary
Total      Info(0-4)      Warning(5--19)      Error(20-29)      Severe(30-39)      Terminal(40-99)
   8         3             5                 0                 0                 0
Source records read . . . . . : 149
Copy records read . . . . . : 37
Copy members processed . . . . . : 1
Sequence errors . . . . . : 0
Highest severity message issued . . : 10
LBL0901 00 Program CTDBAT created in library ICFLIB.
    * * * * * E N D   O F   C O M P I L A T I O N   * * * * *

```

Figure 10-11 (Part 4 of 4). Target Program Example — CTDBAT (User-Defined Formats)

```

5738CB1 V2R1M0 910524          IBM AS/400 COBOL/400          ICFLIB/CTFBAT          RCHAS400 10/03/90 14:35:31    Page 1
Program . . . . . : CTFBAT
Library . . . . . : ICFLIB
Source file . . . . . : QICFPUB
Library . . . . . : ICFLIB
Source member . . . . . : CTFBAT 02/27/89 09:38:46
Generation severity level . . . . . : 29
Text 'description' . . . . . : cobol batch file transfer using $$FORMAT (target)
Source listing options . . . . . : *SOURCE
Generation options . . . . . : *NONE
Message limit:
  Number of messages . . . . . : *NOMAX
  Message limit severity . . . . . : 29
Print file . . . . . : QSYSPRT
Library . . . . . : *LIBL
FIPS flagging . . . . . : *NOFIPS *NOSEG *NODEB *NOOBSOLETE
SAA flagging . . . . . : *NOFLAG
Flagging severity . . . . . : 0
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Compiler . . . . . : IBM AS/400 COBOL/400
5738CB1 V2R1M0 910524          AS/400 COBOL Source          ICFLIB/CTFBAT          RCHAS400 10/03/90 14:35:31    Page 2

```

```

STMT SEQNBR -A 1 B. ....2....3....4....5....6....7..IDENTFCN S COPYNAME CHG DATE
1 000100 IDENTIFICATION DIVISION.                                02/21/89
2 000200 PROGRAM-ID. CTFBAT.                                     02/21/89
000300*****
000400* THIS TARGET PROGRAM IS EVOKED BY THE SOURCE PROGRAM AND * 02/22/89
000500* RECEIVES RECORDS FROM IT. WHEN THE SOURCE PROGRAM IS DONE * 02/22/89
000600* SENDING DATA, THIS PROGRAM SENDS ITS OWN RECORDS UNTIL IT IS * 02/22/89
000700* DONE. WHEN THIS PROGRAM IS DONE, IT SENDS A DETACH REQUEST TO * 02/22/89
000800* THE SOURCE PROGRAM AND ENDS ITS SESSION AND JOB. * 02/22/89
000900*****
3 001000 ENVIRONMENT DIVISION.                                    10/16/87
4 001100 CONFIGURATION SECTION.                                  10/16/87
5 001200 SOURCE-COMPUTER. IBM-AS400.                             02/21/89
6 001300 OBJECT-COMPUTER. IBM-AS400.                             02/21/89
7 001400 SPECIAL-NAMES. I-O-FEEDBACK IS FEEDBACK-AREA           10/16/87
8 001500 OPEN-FEEDBACK IS OPEN-FBA.                             10/16/87
9 001600 INPUT-OUTPUT SECTION.                                   10/16/87
001700* 1
10 001800 FILE-CONTROL.                                         10/16/87
11 001900 SELECT DBFILE ASSIGN TO DATABASE-DBFILE.              10/16/87
12 002000 SELECT TGTICF ASSIGN TO WORKSTATION-TGTICF-SI         10/16/87
13 002100 ORGANIZATION IS TRANSACTION                            10/16/87
14 002200 FILE STATUS IS STATUS-IND MAJ-MIN.                    10/16/87
15 002300 SELECT DFILE ASSIGN TO DATABASE-HEXDUMP.              10/16/87
16 002400 SELECT QPRINT ASSIGN TO PRINTER-QSYSPRT.             10/16/87
17 002500 DATA DIVISION.                                       10/16/87
18 002600 FILE SECTION.                                         10/16/87
19 002700 FD DBFILE                                             10/16/87
20 002800 LABEL RECORDS ARE STANDARD.                           10/16/87
21 002900 01 PREC.                                              10/16/87
22 003000 05 PREC-DATA PIC X(80).                                10/16/87
23 003100 FD TGTICF                                             10/16/87
24 003200 LABEL RECORDS ARE STANDARD.                           10/16/87
25 003300 01 ICFREC.                                            10/16/87
26 003400 05 SNDDATA-0.                                          10/16/87
27 003500 10 SNDLENGTH PIC 9(4).                                10/16/87
28 003600 10 SNDFIELD PIC X(80).                                10/16/87
29 003700 05 ENDREC-0 REDEFINES SNDDATA-0.                     10/16/87
30 003800 10 ENLENGTH PIC 9(4).                                  10/16/87
31 003900 10 FILLER PIC X(80).                                  10/16/87
32 004000 FD DFILE                                              10/16/87
33 004100 LABEL RECORDS ARE STANDARD.                           10/16/87
34 004200 01 DUMPREC.                                           10/16/87
35 004300 05 DUMP-MAJ-MIN PIC X(4).                              10/16/87
36 004400 05 DUMP-RECORD PIC X(80).                             10/16/87
37 004500 FD QPRINT                                             10/16/87
38 004600 LABEL RECORDS ARE OMITTED.                            10/16/87
39 004700 01 PRINTREC PIC X(132).                               10/16/87
40 004800 WORKING-STORAGE SECTION.                               10/16/87
41 004900 77 MAJ-MIN-SAV PIC X(4).                               10/16/87
42 005000 77 STATUS-IND PIC X(2).                               10/16/87
43 005100 77 INDON PIC 1 VALUE B"1".                             10/16/87
44 005200 77 INDOFF PIC 1 VALUE B"0".                           10/16/87
45 005300 77 LEN PIC 9(10)V9(5) COMP                            10/16/87
46 005400 VALUE 0.                                              10/16/87
47 005500 77 CMD2 PIC X(31).                                    10/16/87

```

Figure 10-12 (Part 1 of Target Program Example — CTFBAT (System-Supplied Formats))

```

5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTFBAT RCHAS400 10/03/90 14:35:31 Page 3
STMT SEQNBR -A 1 B.+. . . . 2. . . . +. . . . 3. . . . +. . . . 4. . . . +. . . . 5. . . . +. . . . 6. . . . +. . . . 7. . IDENTFCN S COPYNAME CHG DATE
48 005600 VALUE "CPYF HEXDUMP *LIST PRPFMT(*HEX)". 10/16/87
49 005700 01 OPEN-FBA. 10/16/87
50 005800 05 FILLER PIC X(75). 10/16/87
51 005900 05 RECS-IN-DB PIC 9(09) COMP-4. 10/16/87
52 006000 05 FILLER PIC X(45). 10/16/87
53 006100 01 MAJ-MIN. 10/16/87
54 006200 05 MAJ PIC X(2). 10/16/87
55 006300 05 MIN PIC X(2). 10/16/87

```

```

5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTFBAT RCHAS400 10/03/90 14:35:31 Page 4
STMT SEQNBR -A 1 B.+. . . . 2. . . . +. . . . 3. . . . +. . . . 4. . . . +. . . . 5. . . . +. . . . 6. . . . +. . . . 7. . IDENTFCN S COPYNAME CHG DATE
006400/ 10/16/87
56 006500 PROCEDURE DIVISION. 10/16/87
006600 DECLARATIVES. 10/16/87
006700 ERR-SECTION SECTION. 10/16/87
006800* 2 10/16/87
006900* 10/16/87
007000 USE AFTER STANDARD ERROR PROCEDURE ON TGTICF. 10/16/87
007100 TGTICF-EXCEPTION. 10/16/87
007200* 02/27/89
007300***** 02/27/89
007400* GET INFORMATION FROM THE MAJOR-MINOR CODE AND PLACE IT INTO * 02/27/89
007500* A DATABASE FILE. THEN PRINT THE FILE IN HEX USING COPYFILE. * 02/27/89
007600***** 02/27/89
007700* 02/27/89
007800 GETFBA. 10/16/87
57 007900 MOVE "CTFBAT HAS COMPLETED ABNORMALLY" TO PRINTREC. 10/16/87
58 008000 WRITE PRINTREC. 10/16/87
59 008100 OPEN OUTPUT DFILE. 10/16/87
60 008200 MOVE MAJ-MIN TO DUMP-MAJ-MIN. 10/16/87
61 008300 MOVE ICFREC TO DUMP-RECORD. 10/16/87
62 008400 WRITE DUMPREC. 10/16/87
63 008500 CLOSE DFILE. 10/16/87
64 008600 MOVE 31 TO LEN. 10/16/87
65 008700 CALL "QCAEXEC" USING CMD2 LEN. 10/16/87
66 008800 STOP RUN. 10/16/87
008900* 10/16/87
009000 EXIT-DECLARATIVES. 10/16/87
009100 EXIT. 10/16/87
009200* 10/16/87
67 009300 END DECLARATIVES. 10/16/87

```

```

5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTFBAT RCHAS400 10/03/90 14:35:31 Page 5
STMT SEQNBR -A 1 B.+. . . . 2. . . . +. . . . 3. . . . +. . . . 4. . . . +. . . . 5. . . . +. . . . 6. . . . +. . . . 7. . IDENTFCN S COPYNAME CHG DATE
009400/ 10/16/87
009500 START-PROGRAM SECTION. 10/16/87
009600 START-PROGRAM-PARAGRAPH. 10/16/87
009700* 3 10/16/87
68 009800 OPEN OUTPUT QPRINT 10/16/87
009900 I-O TGTICF 10/16/87
010000 INPUT DBFILE. 10/16/87
010100* 10/16/87
010200***** 02/22/89
010300* DATA CONTINUES TO BE RECEIVED FROM THE PROGRAM DEVICE UNTIL THE * 02/22/89
010400* RCVTRNRND INDICATOR IS SET. EACH RECORD RECEIVED IS PRINTED TO * 02/22/89
010500* THE PRINT FILE. * 02/22/89
010600***** 02/22/89
010700* 10/16/87
010800 RECEIVE-DATA. 10/16/87
010900* 4 10/16/87
69 011000 READ TGTICF. 10/16/87
70 011100 IF MAJ NOT = "03" 10/16/87
71 011200 MOVE ICFREC TO PRINTREC 10/16/87
72 011300 WRITE PRINTREC. 10/16/87
011400* 5 10/16/87
73 011500 IF MIN = "01" THEN 10/16/87
74 011600 GO TO RECEIVE-DATA. 10/16/87
011700* 10/16/87
011800***** 02/22/89
011900* RECORD IS READ FROM THE DATABASE FILE AND IS SENT TO THE SOURCE * 02/22/89
012000* PROGRAM. DATA TRANSMISSION CONTINUES UNTIL END OF FILE IS * 02/22/89
012100* DETECTED ON THE DATABASE FILE. AT THIS TIME, A DETACH SIGNAL IS * 02/22/89
012200* SENT TO THE SOURCE PROGRAM. * 02/22/89
012300***** 02/22/89
012400* 10/16/87
012500 SEND-DATA. 10/16/87
012600* 6 10/16/87
75 012700 READ DBFILE AT END GO TO SIGNAL-DETACH. 10/16/87
77 012800 MOVE PREC TO SNDFIELD OF SNDDATA-O. 10/16/87
78 012900 MOVE +80 TO SNDLENGTH OF SNDDATA-O. 10/16/87

```

Figure 10-12 (Part 2 of 3). Target Program Example — CTFBAT (System-Supplied Formats)


```

79 013000 WRITE ICFREC FORMAT IS "$$SENDNI". 10/16/87
80 013100 GO TO SEND-DATA. 10/16/87
013200* 10/16/87
013300***** 02/22/89
013400* SIGNAL DETACH TO THE SOURCE PROGRAM. * 02/22/89
013500***** 02/22/89
013600* 10/16/87
013700 SIGNAL-DETACH. 10/16/87
013800* 7 10/16/87
81 013900 MOVE SPACES TO ENDREC-0. 10/16/87
82 014000 MOVE +0 TO ENLENGTH OF ENDREC-0. 10/16/87
83 014100 WRITE ICFREC FORMAT IS "$$SENDET". 10/16/87
014200***** 02/22/89
014300* WHEN THE END OF FILE IS REACHED, AN EOJ MESSAGE IS PRINTED AND * 02/22/89
014400* THE PROGRAM ENDS. * 02/22/89
014500***** 02/22/89
014600* 8 10/16/87
84 014700 MOVE "CTFBAT HAS COMPLETED NORMALLY" TO PRINTREC 10/16/87
85 014800 WRITE PRINTREC. 10/16/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTFBAT RCHAS400 10/03/90 14:35:31 Page 6
STMT SEQNBR -A 1 B..+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
86 014900 CLOSE DBFILE 10/16/87
015000 TGTICF 10/16/87
015100 QPRINT. 10/16/87
87 015200 STOP RUN. 10/16/87
***** END OF SOURCE *****
5738CB1 V2R1M0 910524 AS/400 COBOL Messages ICFLIB/CTFBAT RCHAS400 10/03/90 14:35:31 Page 7
STMT
* 19 MSGID: LBL0650 SEVERITY: 00 SEQNBR: 002700
Message . . . . : Blocking/Deblocking for file 'DBFILE' will
be performed by compiler-generated code.
* 32 MSGID: LBL0650 SEVERITY: 00 SEQNBR: 004000
Message . . . . : Blocking/Deblocking for file 'DFILE' will be
performed by compiler-generated code.
* 57 MSGID: LBL0335 SEVERITY: 00 SEQNBR: 007800
Message . . . . : Empty paragraph or section precedes 'GETFBA'
paragraph or section.
***** END OF MESSAGES *****
Message Summary
Total Info(0-4) Warning(5-19) Error(20-29) Severe(30-39) Terminal(40-99)
3 3 0 0 0 0
Source records read . . . . . : 152
Copy records read . . . . . : 0
Copy members processed . . . . . : 0
Sequence errors . . . . . : 0
Highest severity message issued . . : 0
LBL0901 00 Program CTFBAT created in library ICFLIB.
***** END OF COMPILATION *****

```

Figure 10-12 (Part 3 of 3). Target Program Example — CTFBAT (System-Supplied Formats)

Multiple-Session Inquiry (Example II)

This example illustrates an interactive inquiry application that communicates with multiple ICF sessions. A source AS/400 system program accepts inquiries from a display device and sends a request to one of four AS/400 systems. The source program communicates with the display device through a display file, and with the four remote systems through a single ICF file.

The purpose of this example is to show multiple sessions from a single ICF file. The source

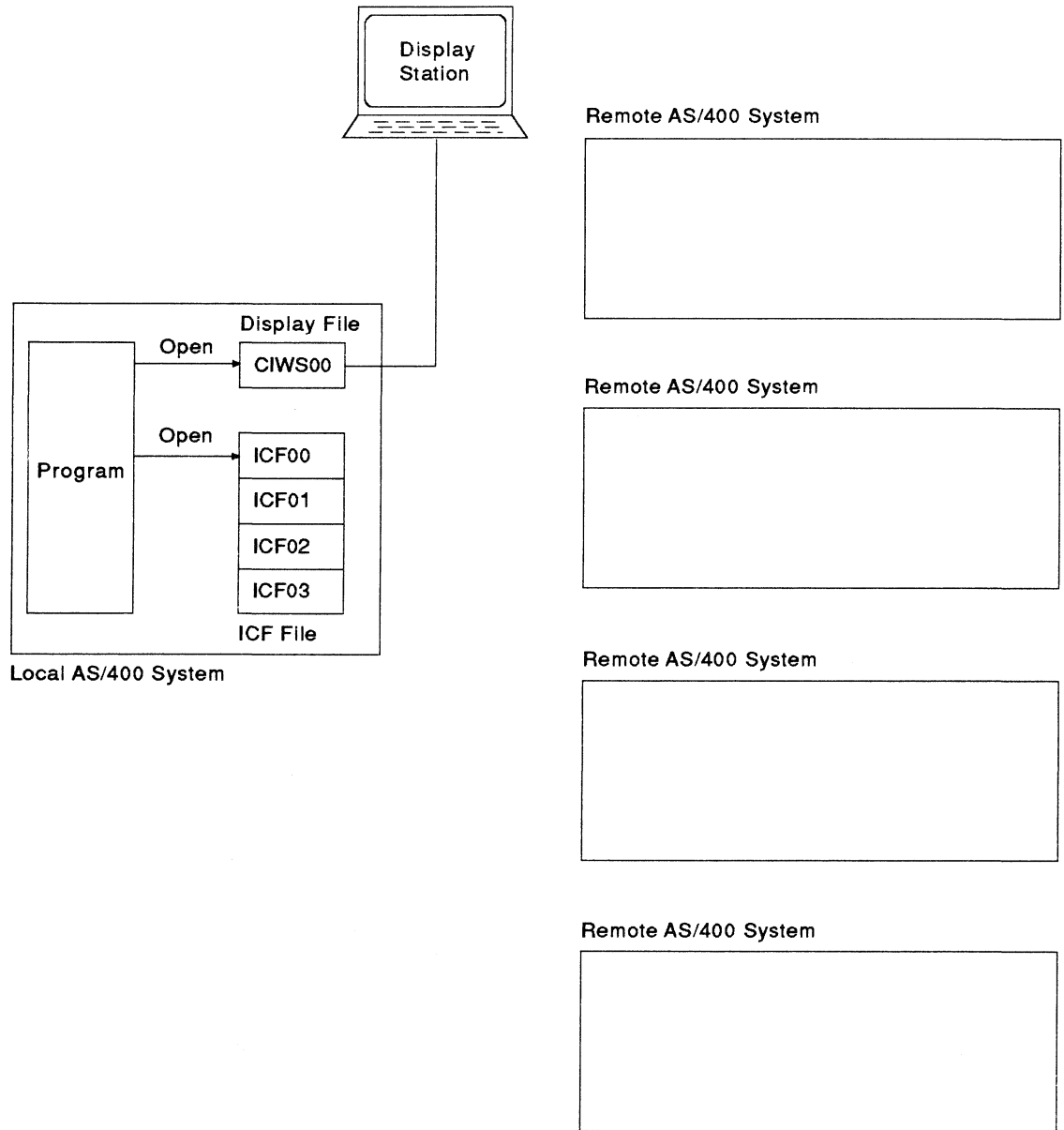
program communicates with four sessions. From the viewpoint of each of the four target programs, there is only one session (with the requesting program device). Therefore, the target programs do not require any unique logic to support the multiple-session source.

Both the source program and the target program are described. The same target program is evoked in each of the four separate remote systems. Therefore, only one target program is shown in the programming example.

Transaction Flow of the Multiple-Session Inquiry (Example II):

The program shown in Figure 10-13 is started from a display station, and both the display and the ICF file are opened. CIWS00 is the *REQUESTER device, acquired when the display file opens. CIWS00 is acquired because DEV(*REQUESTER) was specified when

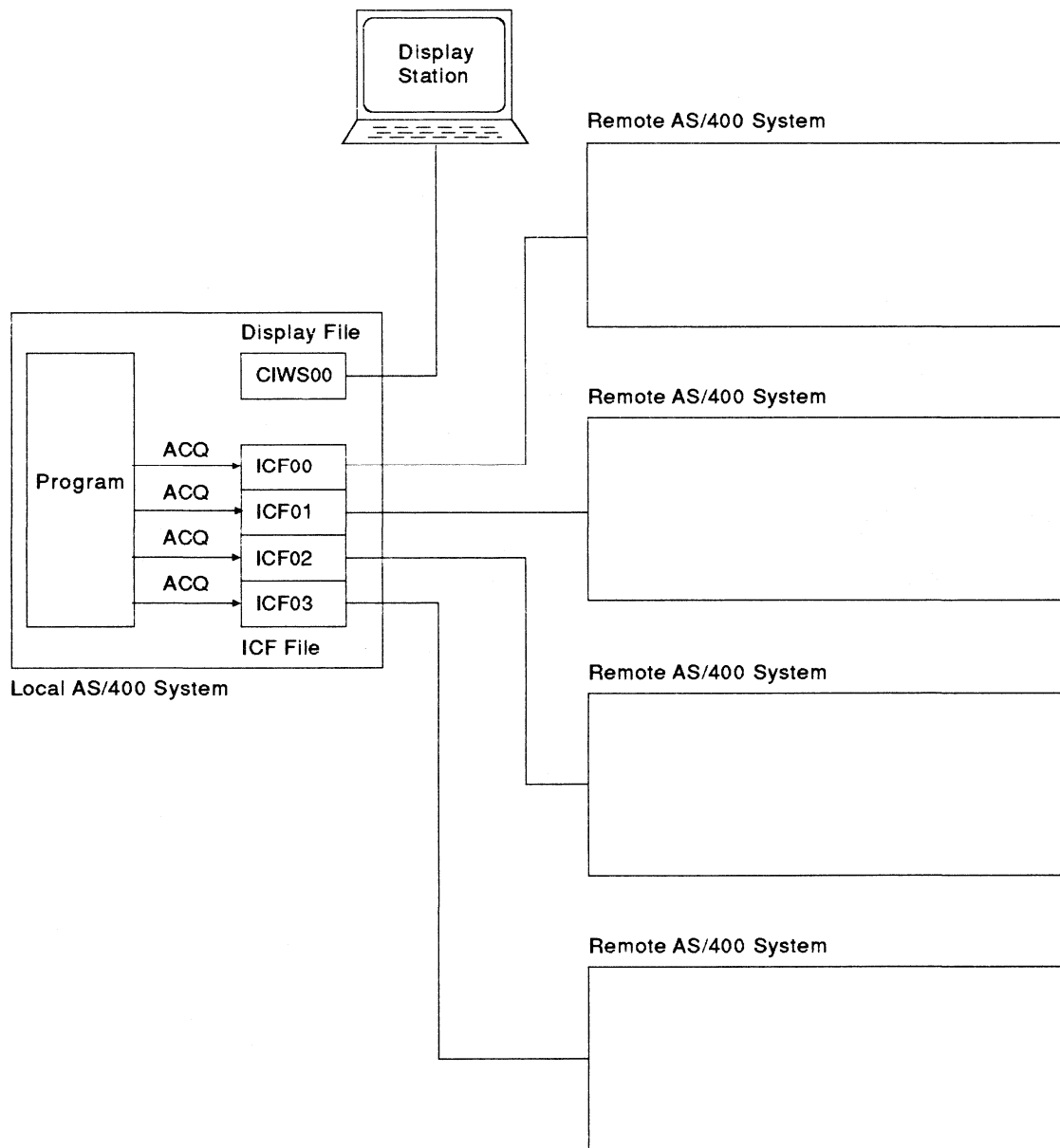
the display file was created. Since the ICF file was created with ACQPGMDEV(*NONE), no ICF program devices are acquired during open processing.



RSL5100-4

Figure 10-13. Program Starts at Display Station

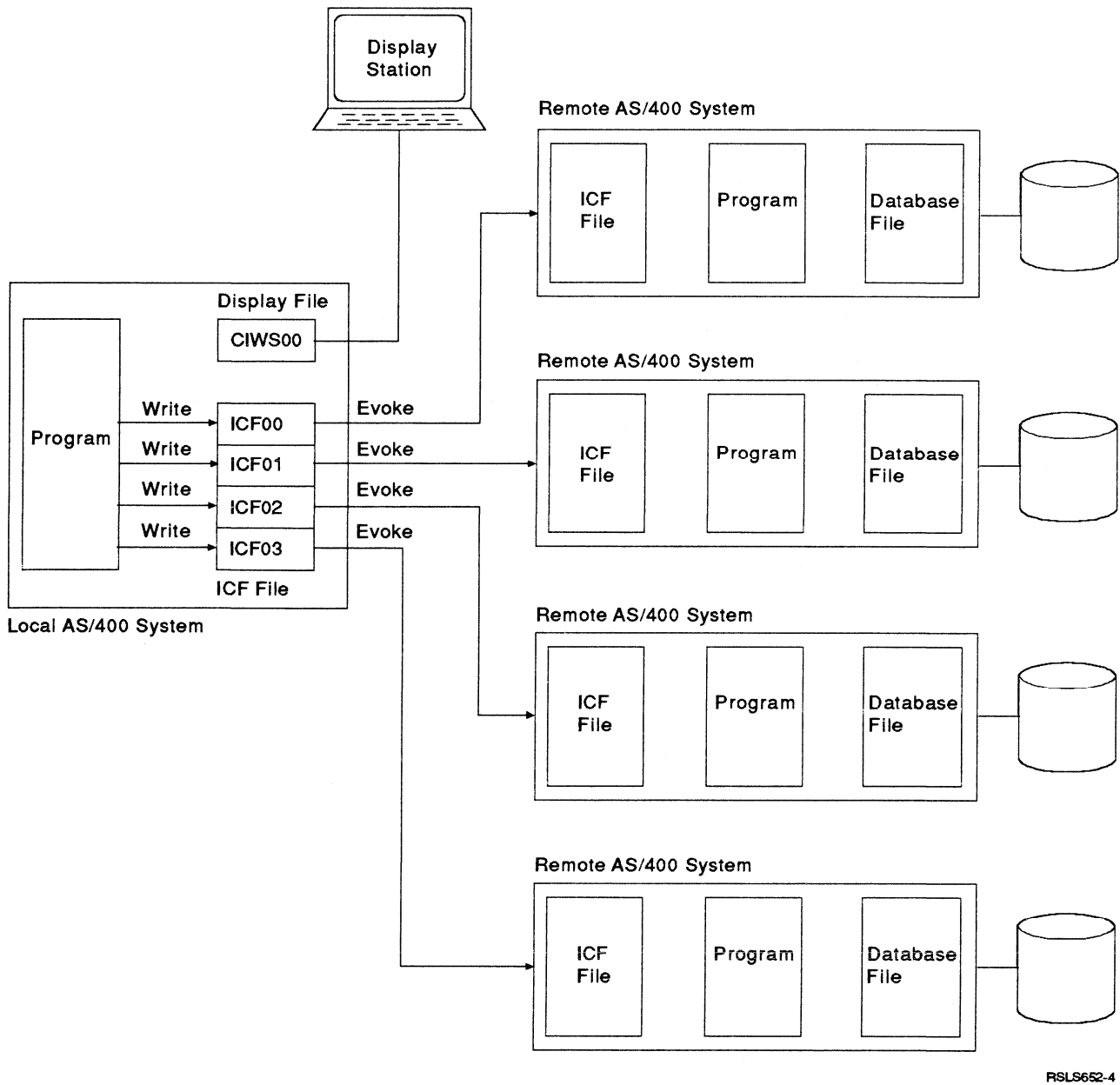
All other program devices must be explicitly acquired by the program, as shown in Figure 10-14.



RSL5651-4

Figure 10-14. Program Devices Explicitly Acquired

All target programs are started with the evoke, as shown in Figure 10-15.

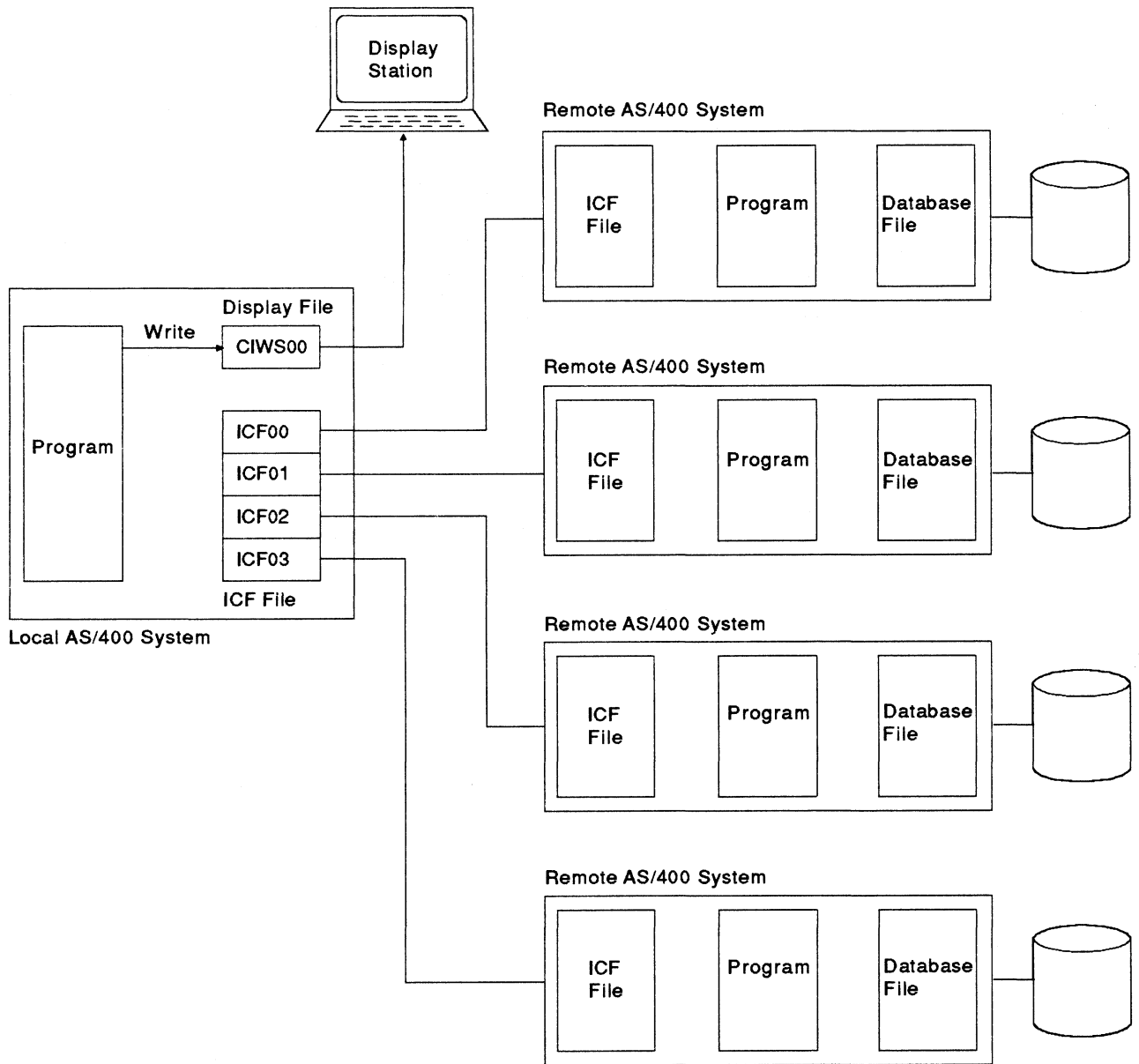


RSL662-4

Figure 10-15. Evoke Starts Target Programs

The source program uses a specific program device name. Each target program uses an ICF file with a program device name that is associated with the requesting program device. The target program's only session is the one used to communicate with the source program. The ICF file on the remote system must be opened by the COBOL language support using the open opera-

tion. Since the file was created with the requesting program device specified on the ACQPGMDEV parameter, the requesting program device is acquired when the file is opened. The main menu is written to the display station on the local system, and the program waits for a request from the display station, as shown in Figure 10-16.

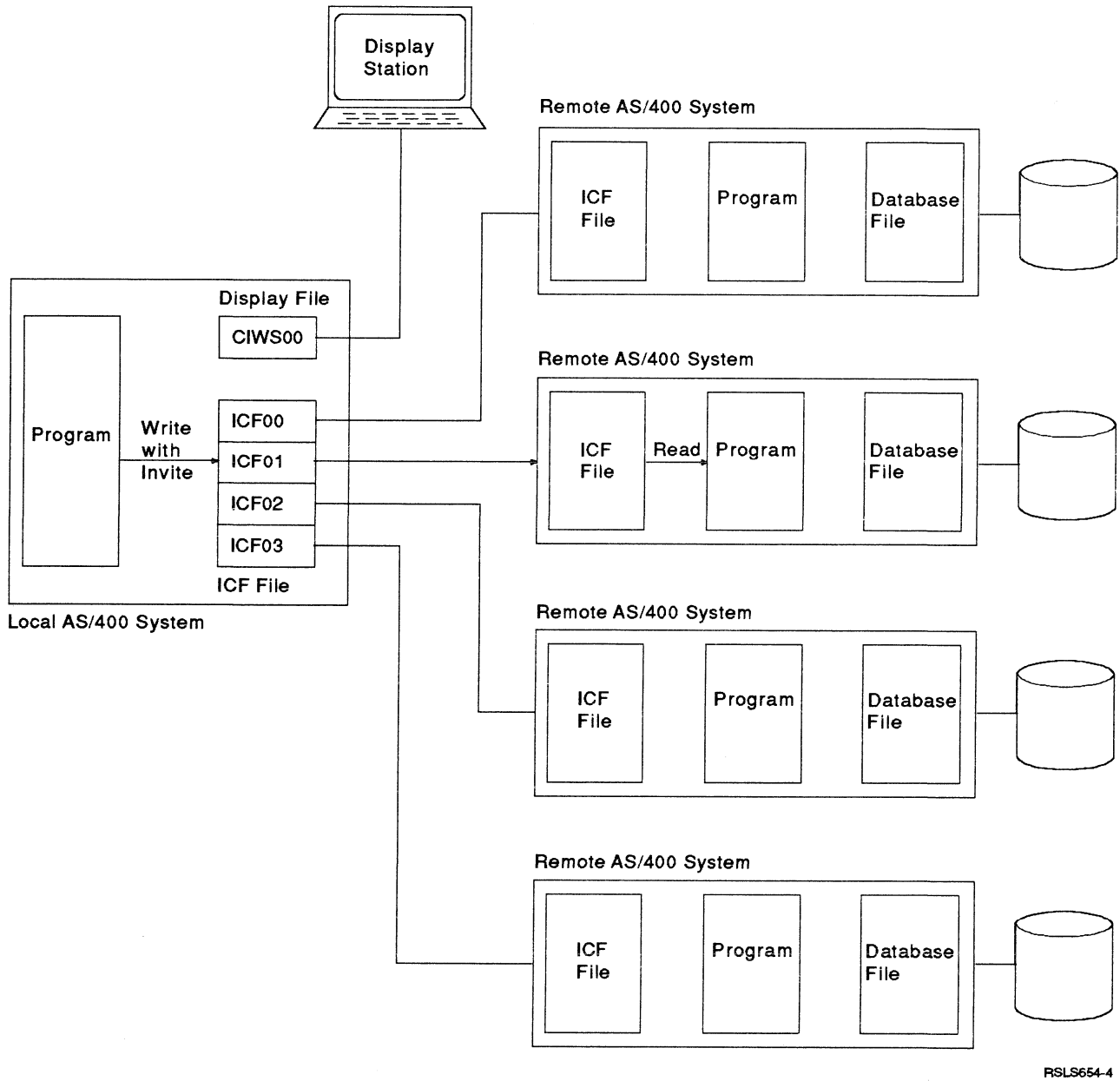


RSL8653-5

Figure 10-16. Main Menu Written to Display Station

The source program sends an inquiry request to one of the remote systems based on the request

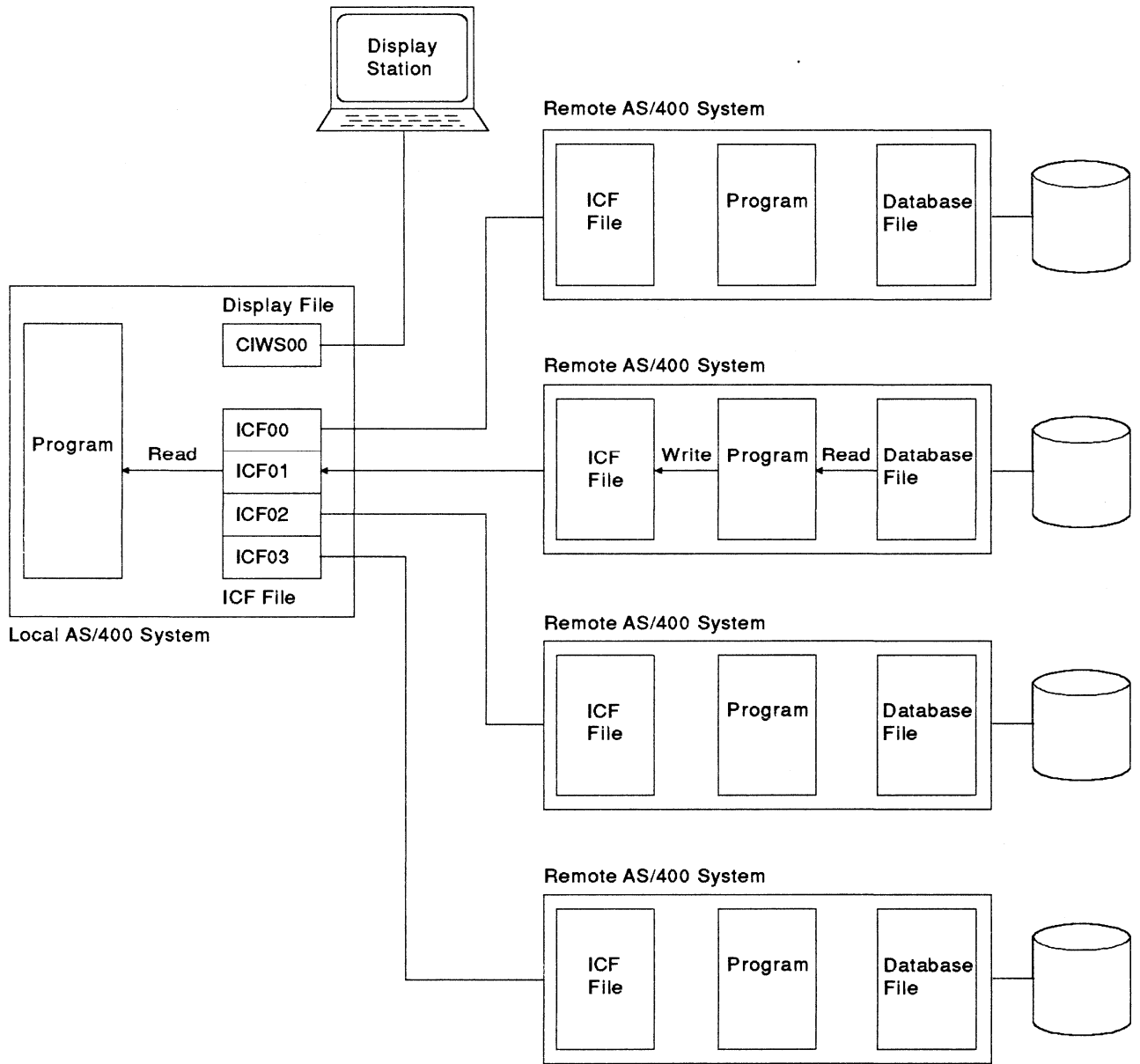
made from the display station, as shown in Figure 10-17.



RSL654-4

Figure 10-17. Program Sends Inquiry Request to Remote System

The target program responds to the inquiry by sending a reply, as shown in Figure 10-18.



RSL655-4

Figure 10-18. Target Program Sends a Reply

The program sends a detach request and ends the session when function key 1 is pressed (while the main inquiry menu is present), as shown in Figure 10-19.

Source Program Multiple-Session Inquiry (Example II):

The following describes a COBOL source program multiple-session inquiry.

Program Files: The COBOL multiple session source program uses the following files:

CMNFIL

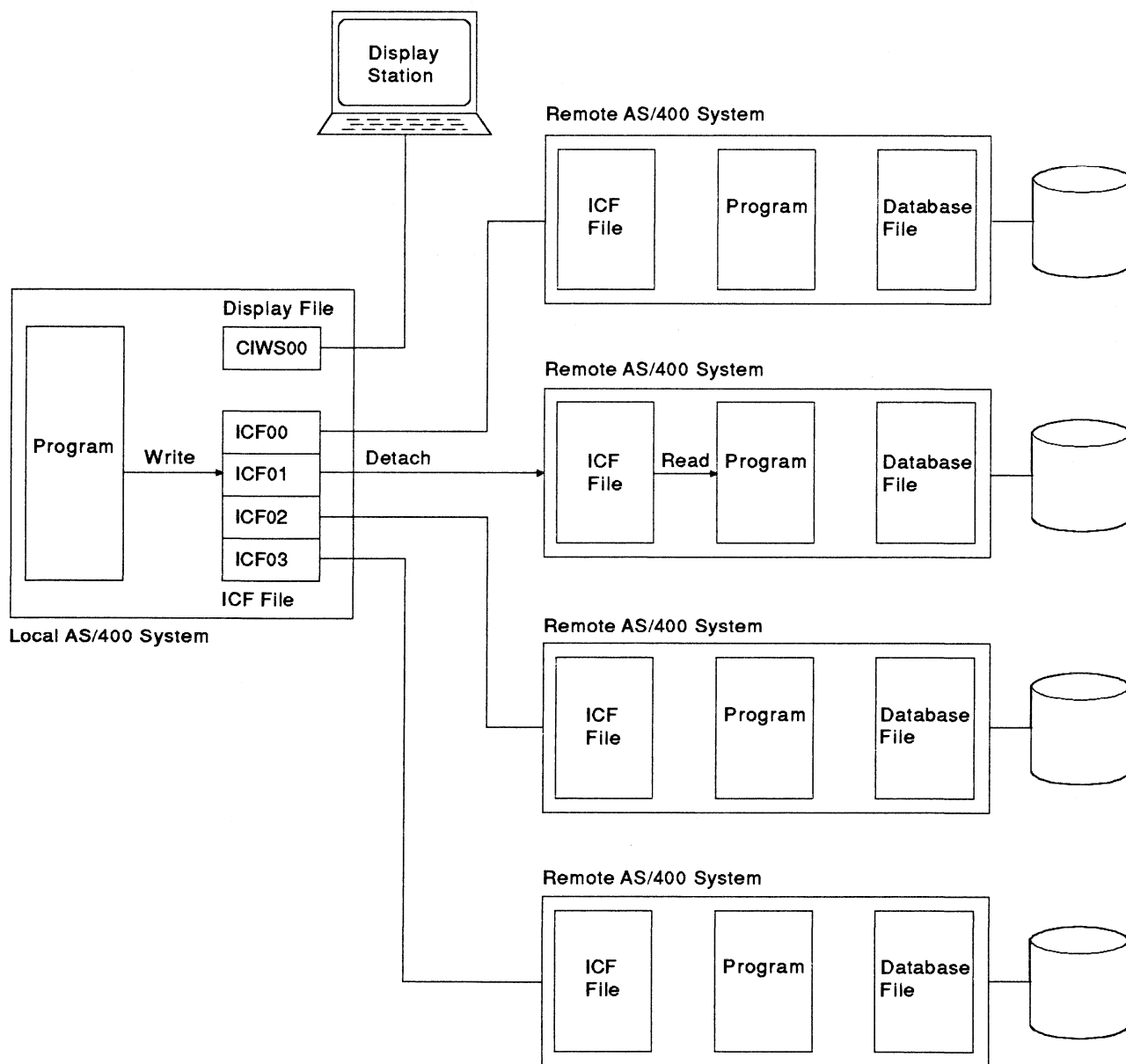
An ICF file used to send records to and receive records from the target program.

DSPFIL

A display file used to enter requests that are to be sent to the target program.

QPRINT

A printer file used to print error messages resulting from communications errors.



RSL5656-5

Figure 10-19. Program Ends the Session

DDS Source: The DDS for the ICF file (CMNFIL) is illustrated in Figure 10-20.

```

5714PW1 R01M00 910524          SEU SOURCE LISTING          10/14/87 17:20:41          PAGE 1
SOURCE FILE . . . . . QICFPUB/ICFLIB
MEMBER . . . . . CMNFIL
SEONBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100                                10/06/87
200                                10/06/87
300      A*****
400      A*                                *
500      A*                                *
600      A*      USED IN SOURCE MULTIPLE SESSION PROGRAM      *
700      A*                                *
800      A*****
900      A      INDARA
1000     A      R ITMRSP
1100     A      RECID(1 'I')
1200     A      RECITM      1
1300     A      ITEMNO      6 0
1400     A      DESC      30
1500     A      QTYLST      7 0
1600     A      QTYOH      7 0
1700     A      QTY00      7 0
1800     A      QTYB0      7 0
1900     A      UNITQ      2
2000     A      PR01      7 2
2100     A      PR05      7 0
2200     A      UFRT      5 2
2300     A      SLSTM      9 2
2400     A      SLSTY      11 2
2500     A      CSTTM      9 2
2600     A      CSTTY      11 2
2700     A      PRO      5 2
2800     A      LOS      9 2
2900     A      FILL1      56
3000     A      R DTLRSP
3100     A      RECID(1 'C')
3200     A      RCVTRNRND(90)
3300     A      RECCUS      1
3400     A      CUSTNO      6 0
3500     A      DNAME      30
3600     A      DLSTOR      6 0
3700     A      DSLSTM      9 0
3800     A      DSPM01      9 0
3900     A      DSPM02      9 0
4000     A      DSPM03      9 0
4100     A      DSTTYD      11 0
4200     A      IDEPT      3 0
4300     A      FILL2      57
4400     A      R DETACH
4500     A      DETACH
4600     A      R EOS
4700     A      EOS
4800     A      R EVKREQ
4900     A      EVOKE(&LIB/&PGMID)
5000     A      PGMID      10A P
5100     A      LIB      10A P
5200     A      R ITMREQ
5300     A      INVITE
5714PW1 R01M00 910524          SEU SOURCE LISTING          10/14/87 17:20:41          PAGE 2
SOURCE FILE . . . . . QICFPUB/ICFLIB
MEMBER . . . . . CMNFIL
SEONBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
5400     A      ITEMNO      6 0
5500     A      R DTLREQ
5600     A      INVITE
5700     A      CUSTNO      6 0
          * * * * * E N D O F S O U R C E * * * * *

```

Figure 10-20. DDS for Source Program Multiple Session Inquiry Using CMNFIL

The DDS for the display file (DSPFIL) is illustrated in Figure 10-21.

```

000100071007 A*****
000200071007 A*
000300071007 A*          DISPLAY FILE
000400071007 A*          USED IN SOURCE MULTIPLE SESSION PROGRAM
000500071007 A*
000600071007 A*****
000700071008 A* BEGINNING MENU
000800071008 A*****
000900071007 A          DSPSIZ(*DS3)
001000071007 A          CF01(99) CF02(90) CF03(97)
001100071007 A          R CIMENU          TEXT('MENU FOR INQUIRY')
001200071007 A          1 34'INQUIRY MENU'
001300071007 A          3 1'Select one of the following:'
001400071007 A          4 3'1. Item inquiry'
001500071007 A          5 3'2. Customer inquiry'
001600071007 A          11 1'Options:'
001700071007 A          OPTION          1N I 11 9'VALUES('1' '2')
001800071008 A          R DTLNMU          19 5DFT('CMD KEY 1 - END ')
001900071008 A          TEXT('CUSTOMER INQUIRY SCREEN 1')
002000071007 A          2 2DFT('ENTER CUSTOMER')
002100071013 A          CUSTNO          6N 01 2 20
002200071008 A          19 5DFT('CMD KEY 1 - END ')
002300071008 A          19 23DFT(' 2 - MAIN MENU ')
002400071008 A*
002500071008 A*****
002600071007 A* CUSTOMER INQUIRY SCREEN
002700071008 A*****
002800071007 A          R DTLSCR          TEXT('CUSTOMER INQUIRY SCR. #2')
002900071007 A          1 30FT('CUST DPT LAST ORD & THIS
003000071007 A          $MTH1 &MTH2 $MTH3
003100071008 A          THIS YTD NAME')
003200071008 A          CUSTN          6N 2 2
003300071007 A          DEPT          3N 0 2 9
003400071007 A          DLSTR          6N 0 2 13
003500071007 A          DLSLM          9N 0 2 22
003600071007 A          DSPM1          9N 0 2 32
003700071007 A          DSPM2          9N 0 2 42
003800071007 A          DSPM3          9N 0 2 52
003900071007 A          DSTYD          11N 0 2 62
004000090321 A          CNAME          5 2 74
004100071008 A          19 5DFT('CMD KEY 1 - END ')
004200071008 A          19 23DFT(' 2 - MAIN MENU ')
004300071007 A*
004400071008 A*****
004500071007 A* ITEM INQUIRY SCREEN
004600071008 A*****
004700071007 A          R ITMMNU          TEXT('ITEM INQUIRY SCREEN ONE')
004800071008 A          2 2DFT('ENTER ITEM NUMBER')
004900071013 A          ITEMNO          6N 01 2 20
005000071008 A          19 5DFT('CMD KEY 1 - END ')
005100071008 A          19 23DFT(' 2 - MAIN MENU ')
005200071008 A*****
005300071008 A* ITEM DISPLAY
005400071008 A*****
005500071007 A          R ITMSC2          TEXT('ITEM INQUIRY SCREEN TWO')
005600071007 A          OVERLAY
005700071007 A          4 2DFT('DESC-')
005800071007 A          DSC          30 4 8
005900071007 A          5 2DFT('QUANTITY AVAILABLE')
006000071007 A          QAVAIL          7N 0 5 25
006100071007 A          6 11DFT('ON HAND')
006200071007 A          QTYH          7N 0 6 25
006300071007 A          7 11DFT('ON ORDER')
006400071007 A          QTYO          7N 0 7 25
006500071007 A          8 11DFT('BACK ORDER')
006600071007 A          QTYB          7N 0 8 25
006700071007 A          9 2DFT('UNIT OF MEASURE')
006800071007 A          UNT          2 9 30
006900071007 A          10 2DFT('PRICE PER UNIT')
007000071007 A          PRI          7Y 2 10 24EDTCDE(3)
007100071007 A          11 8DFT('QUANTITY')
007200071007 A          PRS          7Y 0 11 25EDTCDE(3)
007300071007 A          12 8DFT('FREIGHT')
007400071007 A          UFR          5Y 2 12 26EDTCDE(3)
007500071008 A          13 32DFT('MORE... ')
007600071008 A          19 5DFT('CMD KEY 1 - END ')
007700071008 A          19 23DFT(' 2 - MAIN MENU ')
007800071008 A          19 40DFT(' 3 - ITEM MENU ')

```

Figure 10-21 (Part 1 of 2). DDS for Source Program Multiple Session Inquiry Using DSPFIL

```

007900071008 A*****
008000071008 A* ITEM ADDITIONAL DISPLAY
008100071008 A*****
008200071007 A          R ITMSC3          TEXT('ITEM INQUIRY SCREEN 3 ')
008300071007 A          OVERLAY
008400071007 A          5 2DFT('SALES MONTH')
008500071007 A          SLSM          9Y 2 5 16EDTCDE(1)
008600071007 A          6 8DFT('Y-T-D')
008700071007 A          SLSY          11Y 2 6 14EDTCDE(1)
008800071007 A          7 2DFT('COSTS MONTH')
008900071007 A          CSTM          9Y 2 7 16EDTCDE(1)
009000071007 A          8 8DFT('Y-T-D')
009100071007 A          CSTY          11Y 2 8 14EDTCDE(1)
009200071007 A          9 2DFT('PROFIT PCT')
009300071007 A          PROFIT          5Y 2 9 22EDTCDE(1)
009400071007 A          10 2DFT('LOST SALES')
009500071007 A          LOSTS          9Y 2 10 16EDTCDE(1)
009600071008 A          19 5DFT('CMD KEY 1 - END ')
009700071008 A          19 23DFT(' 2 - MAIN MENU ')
009800071008 A*****
009900071007 A* TIMOUT SCREEN.
010000071008 A*****
010100071007 A          R TIMOUT          TEXT('TIME OUT SCREEN')
010200071007 A          OVERLAY
010300071007 A          20 2DFT('REMOTE SYSTEM TIMED OUT. ENTER
010400071007 A          1 TO TRY AGAIN OR 2 TO END.')
010500071007 A          TIMRSP          1 I 20 61

```

Figure 10-21 (Part 2 of 2). DDS for Source Program Multiple Session Inquiry Using DSPFIL

ICF File Creation and Program Device Entry

Definition: The command needed to create the ICF file is:

```

CRITICFF FILE(ICFLIB/CMNFIL) SRCFILE(ICFLIB/QICFPUB)
SRCMBR(CMNFIL) ACQPGMDEV(*NONE) MAXPGMDEV(4)
WAITRCD(30) TEXT("SOURCE ICF FILE FOR MULTIPLE
SESSION PROGRAM")

```

The commands needed to define the four program device entries are:

```

OVRICFDEVE PGMDEV(ICF00) RMTLOCNAME(CHICAGO) FMTSLT(*RECID)
OVRICFDEVE PGMDEV(ICF01) RMTLOCNAME(NEWYORK) FMTSLT(*RECID)
OVRICFDEVE PGMDEV(ICF02) RMTLOCNAME(DETROIT) FMTSLT(*RECID)
OVRICFDEVE PGMDEV(ICF03) RMTLOCNAME(MADISON) FMTSLT(*RECID)

```

Program Explanation: The following explains the structure of the program examples illustrated in Figure 10-22 on page 10-38 and Figure 10-23 on page 10-50. The ICF file used in the first example is defined by the user, and uses externally described data formats (DDS). The second example uses the same file, but uses program-described data and system-supplied formats. The reference numbers in the explanation below correspond to the numbers in the following program examples.

In the following examples, the ICF file used in the first example is externally described, whereas the ICF file used in the second example is a program-described file.

Although the basic structure of the two examples provided is the same, there are differences because of the way the user-defined formats and the system-supplied formats are used. All

output operations to the ICF file in the first example are done using the WRITE statement with the record format name coded as an operand. The output operations to the ICF file in the second example using system-supplied formats are issued with the system-supplied format coded as literal operand.

Differences between the first and second example are described in each of the following descriptions as necessary.

- 1** This section defines the ICF file (CMNFIL) and the display file (DSPFIL) used in the program.

CMNFIL is the ICF file used to send records to and receive records from each of the four target programs. CMNFIL is implemented with the file-level keyword, INDARA, indicating a separate indicator area is used.

DSPFIL is the display file used to receive user's requests and to report the information received based on the request.

The control area clause in the select statements of CMNFIL and DSPFIL is used to define the I/O feedback area. Information from the I/O feedback is used to determine the major/minor return code, record format, and command key pressed.

Note: In the program using system-supplied formats, the input records for CMNFIL are explicitly coded in the program since CMNFIL is now treated as a program-described file. The system-supplied file, QICDMF, could have been used instead of CMNFIL. Using the system-supplied file can be done by specifying QICDMF in the file specification, or by using an OVRICFF command to change the name from CMNFIL to QICDMF.

- 2** DSP-ERROR SECTION and CMN-ERROR SECTION define the error handling procedures for I/O errors on the DSPFIL and CMNFIL. A DSPFIL I/O error causes the program to end, and an error message to be sent to the print file. The section for CMNFIL file I/O errors checks the major/minor return code to determine if the error is recoverable. If the error is recoverable (major code 83), it sets a flag (ERR-SW) to 1 and returns to the program. Furthermore, when major/minor code 3431

(input data truncated) is received, it is saved but not considered as an error, and takes an exit.

- 3** The program opens the files to be used and initializes the ICF file separate indicator area.

- 4** If the ERR-SW switch is set to 1, indicating that a recoverable error has occurred, the program determines whether the open retry count limit nine has been exceeded. If it has, the program goes to **19** and then ends. If the limit count is less than nine, one is added to the count and control passes to **17** and then to **3** to try to open the file.

- 5** The four program devices used by the program are explicitly acquired.

The device for the work station is implicitly acquired when the DSPFIL file is opened.

Also, the evoke requests are issued to the remote systems by passing control to **16**.

When control returns from **16**, the main menu (record format CIMENU) is then written to the work station.

- 6** A read operation is issued to the display device, and the program waits for an input request from the user. When a record is returned, the last record format used (as specified in the RCD-FMT field in the I/O control area) is checked. Based on the value in RCD-FMT, the program branches to the appropriate routine.

If a match is not found for the display record format, the main menu (CIMENU) is written to the work station and control is returned to **6**.

- 7** This routine is called if the request is made from the main menu (CIMENU). If the CMD-KEY variable is set to '01', indicating that the operator pressed command key 1, the four transactions and sessions are ended and the program ends. If the operator entered option 1, the program writes the Item Inquiry menu (ITMMNU) to the work station and returns to **6**.

If the option is not 1, the Customer Inquiry menu (DTLMNU) is written to the work station and control is passed to **6**.

The rest of this chapter discusses the details of the control is passed to **6**.

8 This routine is called when the user is requesting an item inquiry (record format ITMMNU). If command key 1 (CMD-KEY = '01') is pressed, control passes to **19**, and then to **20**, the four transactions end, and the program ends. If command key 2 is pressed, the inquiry request is canceled, the main menu (CMENU) is written to the work station, and the program returns to **16**.

The item number read from the work station is checked for value range. If the range is from 0 to 399999, then the request is sent to the target program on program device ICF01.

If the range is from 400000 to 699999, the request is sent to the target program on program device ICF02.

If the range is from 700000 to 899999, the request is sent to the target program on program device ICF03.

The request is sent to the appropriate target program by writing data to the program device using format ITMREQ. The INVITE keyword is specified as part of the ITMREQ format to give the target program permission to send.

A read request is issued to the program device to receive the response to the inquiry.

The read is an implied read-from-invited-program-devices because no record format is specified in the read statement.

Control goes to **9** to process the item information based on the input data received and the result written to the screen using format ITMSC2.

After returning from **9**, the program returns to **6**.

Note: In the program using system-supplied formats, the \$\$\$SEND format is used as a literal instead of the user-defined ITMREQ record format name.

9 This routine is called when the target program responds to a request for an item record. If the returned item number is 0 or less, the request is invalid and a new Item Inquiry menu (ITMMNU) is written to the work station.

The program then performs the calculations to set the quantity fields and writes the result to the requesting work station using record format ITMSC2.

The program then returns to the calling routine.

10 This routine is called to process the next user request. If command key 1 (CMD-KEY = '01') is pressed, the transactions and session are ended **19**, and control goes to **20** to end the program.

If command key 2 is pressed, the main menu (CMENU) is written to the work station. If command key 3 is pressed, the Item Inquiry menu is written to the work station, and the program returns to **6**. By pressing Enter, the profit and loss figures are calculated and written to the work station before returning control to **6**.

11 This routine calculates the profit and loss figures for the second screen of the requested item number.

12 This routine is called when a request is read from the Customer Inquiry menu (DTLMNU). If command key 1 (CMD-KEY = '01') is pressed, the transactions and sessions are ended. If command key 2 (CMD-KEY = '02') is pressed, the main menu (CIMENU) is written to the work station and the program returns to **6**.

The customer inquiry request is sent to the target program by writing data to the program device ICF00 using format DTLREQ. The INVITE keyword is specified as part of the DLTREQ format to give the target program permission to send.

Control goes to **13** to retrieve the customer detail information.

Routine **14** is called to continue the customer information processing.

The program returns to **6**.

Note: In the program using system-supplied formats, the write operation is issued with the \$\$\$SEND format specified as literal, in the user-supplied format, WRITE was issued with a record format name DTLREQ.

13 The information supplied by the target program in response to a request for a customer detail is processed in this routine. If

the customer number is 0 or less, the request is invalid and the main menu (record format CIMENU) is written to the work station. The program then returns to **6**.

Control goes to **15** to retrieve the customer detail information, and the result is written to the work station using record format DTLSCR.

The program then returns to **6**.

14 This routine is called from **6** and handles the user's request following the display of the customer information. Command key 1 ends the job, command key 2 displays the main menu (CMENU), and pressing Enter displays the Customer Inquiry menu (DTLMNU). Then, control returns to **6**.

15 This routine issues the read operation to the program device.

This read is an implied read-from-invited-program-devices because no record format is specified on the read statement.

A check is made of the MAJ-MIN return code for possible error conditions on a successful return (control is automatically passed to **2** for non-successful I/O operations). A 0310 major-minor return code means the remote system has timed out. (The wait time was specified on the CRTICFF command.) If no data was received (MAJ-MIN - 03xx), the request is sent again to the remote system. Finally, if the data returns in the wrong format, Control is passed to **17**.

The customer information received from the target program is processed, and the result is written to the user work station using screen format DTLBLK.

Control returns to the calling routine.

Note: The program using system-supplied formats issues the READ statement with the file name CMNFIL specified in the operand without a record format name.

16 This routine builds the evoke requests to send to the remote systems. Because the DDS keyword for the record format only specifies the field identifiers with the record, this code moves the literal value CTDMULCL to the field *PGMID*, and ICFLIB to the field *LIB*.

When the program start request is received at the remote system, ICFLIB is searched for CTDMULCL and that program then starts. CTDMULCL is a CL program that contains the following statements:

```
ADDLIBLE ICFLIB  
CALL ICFLIB/CTDMUL
```

Note: In the program using system-supplied formats, the evoke request is issued using the WRITE statement followed with a \$\$EVOKNI format coded as literal in the operand.

The library and program (ICFLIB/CTFMULCL) are specified as part of the \$\$EVOKNI format. CTFMULCL is a CL program that contains the following statements:

```
ADDLIBLE ICFLIB  
CALL ICFLIB/CTFMUL
```

17 This routine ends the transactions and closes the files. The ERR-SW indicator is set again, and control returns to the calling routine.

18 This routine is executed when the program detects data in an incorrect record format. It writes an error message to the print file, ends the program, and implicitly ends the session.

19 This routine issues the detach function to the ICF file for each of the four program devices. In the program using the system-supplied format, the write operation is issued using \$\$SENDET format, but in the program using the user-supplied format, it is the record format name DETACH.

20 This routine releases the program devices and close the files. The program ends.

```

5738CB1 V2R1M0 910524          IBM AS/400 COBOL/400          ICFLIB/CSDMUL          RCHAS400 10/03/90 14:33:13      Page 1
Program . . . . . : CSDMUL
Library . . . . . : ICFLIB
Source file . . . . . : QICFPUB
Library . . . . . : ICFLIB
Source member . . . . . : CSDMUL 10/03/90 14:28:28
Generation severity level . . . . . : 29
Text 'description' . . . . . : CBL Multiple Session Inquiry - Source DDS
Source listing options . . . . . : *SOURCE
Generation options . . . . . : *NONE
Message limit:
  Number of messages . . . . . : *NOMAX
  Message limit severity . . . . . : 29
Print file . . . . . : QSYSPRT
Library . . . . . : *LIBL
FIPS flagging . . . . . : *NOFIPS *NOSEG *NODEB *NOOBSOLETE
SAA flagging . . . . . : *NOFLAG
Flagging severity . . . . . : 0
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Compiler . . . . . : IBM AS/400 COBOL/400
5738CB1 V2R1M0 910524          AS/400 COBOL Source          ICFLIB/CSDMUL          RCHAS400 10/03/90 14:33:13      Page 2
STMT SEQNBR -A 1 B.+. . . . . 2. . . . . 3. . . . . 4. . . . . 5. . . . . 6. . . . . 7. IDENTFCN S COPYNAME CHG DATE
1 000100 IDENTIFICATION DIVISION.                                09/30/87
2 000200 PROGRAM-ID. CSDMUL.                                     09/30/87
000300*****
000400* THIS PROGRAM ASSIGNS FOUR SESSIONS AS FOLLOWS: *          09/30/87
000500* 'ICF00' TO INQUIRE ABOUT A CUSTOMER ACCOUNT BEFORE AN *          09/30/87
000600* ORDER IS PROCESSED. *          09/30/87
000700* 'ICF01' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM *          09/30/87
000800* BEING ORDERED (ITEM 000001 THRU 399999). *          09/30/87
000900* 'ICF02' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM *          09/30/87
001000* BEING ORDERED (ITEM 400000 THRU 699999). *          09/30/87
001100* 'ICF03' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM *          09/30/87
001200* BEING ORDERED (ITEM 700000 THRU 999999). *          09/30/87
001300* A DISPLAY DEVICE IS USED TO ENTER THE REQUEST ( USING A *          09/30/87
001400* CUSTOMER AND AN ITEM MENU) THAT IS SENT TO THE REMOTE *          09/30/87
001500* SYSTEM. *          10/15/87
001600*****
3 001700 ENVIRONMENT DIVISION.                                09/30/87
4 001800 CONFIGURATION SECTION.                                09/30/87
5 001900 SOURCE-COMPUTER. IBM-AS400.                            01/15/88
6 002000 OBJECT-COMPUTER. IBM-AS400.                            01/15/88
7 002100 SPECIAL-NAMES. I-O-FEEDBACK IS I-O-FEEDBACK          09/30/87
8 002200 OPEN-FEEDBACK IS OPEN-FBA.                            09/30/87
9 002300 INPUT-OUTPUT SECTION.                                09/30/87
10 002400 FILE-CONTROL.                                       09/30/87
002500* 1
002600*****
002700* *          09/30/87
002800* FILE SPECIFICATIONS *          09/30/87
002900* *          09/30/87
003000* CMNFIL : ICF FILE USED TO SEND A REQUEST TO ONE *          10/03/90
003100* OF FOUR DIFFERENT TARGET PROGRAMS. MULTIPLE *          09/30/87
003200* SESSIONS ARE ACTIVE CONCURRENTLY. *          09/30/87
003300* *          09/30/87
003400* DSPFIL : DISPLAY FILE USED TO ENTER A REQUEST TO BE *          09/30/87
003500* SENT TO A REMOTE SYSTEM. *          09/30/87
003600* *          09/30/87
003700*****
11 003800 SELECT CMNFIL ASSIGN TO WORKSTATION-CMNFIL-SI          10/13/87
12 003900 ORGANIZATION IS TRANSACTION                          09/30/87
13 004000 CONTROL-AREA IS TR-CTL-AREA                          09/30/87
14 004100 FILE STATUS IS STATUS-IND MAJ-MIN.                  09/30/87
15 004200 SELECT DSPFIL ASSIGN TO WORKSTATION-DSPFIL          09/30/87
16 004300 ORGANIZATION IS TRANSACTION                          09/30/87
17 004400 CONTROL-AREA IS DISPLAY-FEEDBACK                    09/30/87
18 004500 FILE STATUS IS STATUS-DSP.                          09/30/87
19 004600 SELECT QPRINT ASSIGN TO PRINTER-QSYSPRT.            09/30/87
20 004700 DATA DIVISION.                                       09/30/87
21 004800 FILE SECTION.                                         09/30/87
22 004900 FD CMNFIL                                           09/30/87
23 005000 LABEL RECORDS ARE STANDARD.                          09/30/87
24 005100 01 CMNREC.                                           09/30/87
25 005200 COPY DDS-ALL-FORMATS-I-O OF CMNFIL.                  02/28/89
26 +000001 05 CMNFIL-RECORD PIC X(196).
+000002* I-O FORMAT:ITMRSP FROM FILE CMNFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000003* <-ALL-FMTS
<-ALL-FMTS

```

Figure 10-22 (Part 1 of 12). Source Program Example — CSDMUL (User-Defined Formats)

```

5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 3
STMT SEQNBR -A 1 B..+....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
27 +000004 05 ITMRSP REDEFINES CMNFIL-RECORD. <-ALL-FMTS
28 +000005 06 RECITM PIC X(1). <-ALL-FMTS
29 +000006 06 ITEMNO PIC S9(6). <-ALL-FMTS
30 +000007 06 DESC PIC X(30). <-ALL-FMTS
31 +000008 06 QTYLST PIC S9(7). <-ALL-FMTS
32 +000009 06 QTYOH PIC S9(7). <-ALL-FMTS
33 +000010 06 QTYO0 PIC S9(7). <-ALL-FMTS
34 +000011 06 QTYB0 PIC S9(7). <-ALL-FMTS
35 +000012 06 UNITQ PIC X(2). <-ALL-FMTS
36 +000013 06 PR01 PIC S9(5)V9(2). <-ALL-FMTS
37 +000014 06 PR05 PIC S9(7). <-ALL-FMTS
38 +000015 06 UFRT PIC S9(3)V9(2). <-ALL-FMTS
39 +000016 06 SLSTM PIC S9(7)V9(2). <-ALL-FMTS
40 +000017 06 SLSTY PIC S9(9)V9(2). <-ALL-FMTS
41 +000018 06 CSTTM PIC S9(7)V9(2). <-ALL-FMTS
42 +000019 06 CSTTY PIC S9(9)V9(2). <-ALL-FMTS
43 +000020 06 PRO PIC S9(3)V9(2). <-ALL-FMTS
44 +000021 06 LOS PIC S9(7)V9(2). <-ALL-FMTS
45 +000022 06 FILL1 PIC X(56). <-ALL-FMTS
+000023* INPUT FORMAT:DTLRSP FROM FILE CMNFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000024* <-ALL-FMTS
46 +000025 05 DTLRSP-I REDEFINES CMNFIL-RECORD. <-ALL-FMTS
47 +000026 06 RECCUS PIC X(1). <-ALL-FMTS
48 +000027 06 CUSTNO PIC S9(6). <-ALL-FMTS
49 +000028 06 DNAME PIC X(30). <-ALL-FMTS
50 +000029 06 DLSTOR PIC S9(6). <-ALL-FMTS
51 +000030 06 DSLSTM PIC S9(9). <-ALL-FMTS
52 +000031 06 DSPM01 PIC S9(9). <-ALL-FMTS
53 +000032 06 DSPM02 PIC S9(9). <-ALL-FMTS
54 +000033 06 DSPM03 PIC S9(9). <-ALL-FMTS
55 +000034 06 DSTTYD PIC S9(11). <-ALL-FMTS
56 +000035 06 IDEPT PIC S9(3). <-ALL-FMTS
57 +000036 06 FILL2 PIC X(57). <-ALL-FMTS
+000037* OUTPUT FORMAT:DTLRSP FROM FILE CMNFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000038* <-ALL-FMTS
58 +000039 05 DTLRSP-0 REDEFINES CMNFIL-RECORD. <-ALL-FMTS
59 +000040 06 RECCUS PIC X(1). <-ALL-FMTS
60 +000041 06 CUSTNO PIC S9(6). <-ALL-FMTS
61 +000042 06 DNAME PIC X(30). <-ALL-FMTS
62 +000043 06 DLSTOR PIC S9(6). <-ALL-FMTS
63 +000044 06 DSLSTM PIC S9(9). <-ALL-FMTS
64 +000045 06 DSPM01 PIC S9(9). <-ALL-FMTS
65 +000046 06 DSPM02 PIC S9(9). <-ALL-FMTS
66 +000047 06 DSPM03 PIC S9(9). <-ALL-FMTS
67 +000048 06 DSTTYD PIC S9(11). <-ALL-FMTS
68 +000049 06 IDEPT PIC S9(3). <-ALL-FMTS
69 +000050 06 FILL2 PIC X(57). <-ALL-FMTS
+000051* I-0 FORMAT:DETACH FROM FILE CMNFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000052* <-ALL-FMTS
+000053* 05 DETACH REDEFINES CMNFIL-RECORD. <-ALL-FMTS
+000054* I-0 FORMAT:E0S FROM FILE CMNFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000055* <-ALL-FMTS
+000056* 05 E0S REDEFINES CMNFIL-RECORD. <-ALL-FMTS
+000057* INPUT FORMAT:EVKREQ FROM FILE CMNFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000058* <-ALL-FMTS
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 4
STMT SEQNBR -A 1 B..+....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
+000059* 05 EVKREQ-I REDEFINES CMNFIL-RECORD. <-ALL-FMTS
+000060* OUTPUT FORMAT:EVKREQ FROM FILE CMNFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000061* <-ALL-FMTS
70 +000062 05 EVKREQ-0 REDEFINES CMNFIL-RECORD. <-ALL-FMTS
71 +000063 06 PGMID PIC X(10). <-ALL-FMTS
72 +000064 06 LIB PIC X(10). <-ALL-FMTS
+000065* I-0 FORMAT:ITMREQ FROM FILE CMNFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000066* <-ALL-FMTS
73 +000067 05 ITMREQ REDEFINES CMNFIL-RECORD. <-ALL-FMTS
74 +000068 06 ITEMNO PIC S9(6). <-ALL-FMTS
+000069* I-0 FORMAT:DTLREQ FROM FILE CMNFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000070* <-ALL-FMTS
75 +000071 05 DTLREQ REDEFINES CMNFIL-RECORD. <-ALL-FMTS
76 +000072 06 CUSTNO PIC S9(6). <-ALL-FMTS
77 005300 FD DSPFIL 09/30/87
78 005400 LABEL RECORDS ARE STANDARD. 09/30/87
79 005500 01 DSPREC. 09/30/87
80 005600 COPY DDS-ALL-FORMATS-I-0 OF DSPFIL. 02/27/89
81 +000001 05 DSPFIL-RECORD PIC X(79). <-ALL-FMTS
+000002* INPUT FORMAT:CIMENU FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000003* MENU FOR INQUIRY <-ALL-FMTS

```

Figure 10-22 (Part 2 of 12). Source Program Example — CSDMUL (User-Defined Formats)

```

82 +000004      05 CIMENU-I      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
83 +000005      06 CIMENU-I-INDIC.      <-ALL-FMTS
84 +000006          07 IN99          PIC 1 INDIC 99.      <-ALL-FMTS
85 +000007          07 IN98          PIC 1 INDIC 98.      <-ALL-FMTS
86 +000008          07 IN97          PIC 1 INDIC 97.      <-ALL-FMTS
87 +000009      06 OPTION          PIC X(1).      <-ALL-FMTS
+000010* OUTPUT FORMAT:CIMENU      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000011*          MENU FOR INQUIRY      <-ALL-FMTS
+000012*      05 CIMENU-0      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
+000013* INPUT FORMAT:DTLMNU      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000014*          CUSTOMER INQUIRY SCREEN 1      <-ALL-FMTS
88 +000015      05 DTLMNU-I      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
89 +000016      06 DTLMNU-I-INDIC.      <-ALL-FMTS
90 +000017          07 IN99          PIC 1 INDIC 99.      <-ALL-FMTS
91 +000018          07 IN98          PIC 1 INDIC 98.      <-ALL-FMTS
92 +000019          07 IN97          PIC 1 INDIC 97.      <-ALL-FMTS
93 +000020      06 CUSTN0          PIC S9(6).      <-ALL-FMTS
+000021* OUTPUT FORMAT:DTLMNU      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000022*          CUSTOMER INQUIRY SCREEN 1      <-ALL-FMTS
+000023*      05 DTLMNU-0      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
+000024* INPUT FORMAT:DTLSCR      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000025*          CUSTOMER INQUIRY SCR. #2      <-ALL-FMTS
94 +000026      05 DTLSCR-I      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
95 +000027      06 DTLSCR-I-INDIC.      <-ALL-FMTS
96 +000028          07 IN99          PIC 1 INDIC 99.      <-ALL-FMTS
97 +000029          07 IN98          PIC 1 INDIC 98.      <-ALL-FMTS
98 +000030          07 IN97          PIC 1 INDIC 97.      <-ALL-FMTS
+000031* OUTPUT FORMAT:DTLSCR      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000032*          CUSTOMER INQUIRY SCR. #2      <-ALL-FMTS
99 +000033      05 DTLSCR-0      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
100 +000034      06 CUSTN          PIC X(6).      <-ALL-FMTS
101 +000035      06 DEPT          PIC S9(3).      <-ALL-FMTS
102 +000036      06 DLSTR          PIC S9(6).      <-ALL-FMTS
103 +000037      06 DSLSM          PIC S9(9).      <-ALL-FMTS
5738CB1 V2R1M0 910524      AS/400 COBOL Source      ICFLIB/CSDMUL      RCHAS400 10/03/90 14:33:13      Page 5
STMT SEQNBR -A 1 B.+. . . . . 2. . . . . 3. . . . . 4. . . . . 5. . . . . 6. . . . . 7. IDENTFCN S COPYNAME CHG DATE
104 +000038      06 DSPM1          PIC S9(9).      <-ALL-FMTS
105 +000039      06 DSPM2          PIC S9(9).      <-ALL-FMTS
106 +000040      06 DSPM3          PIC S9(9).      <-ALL-FMTS
107 +000041      06 DSTYD          PIC S9(11).      <-ALL-FMTS
108 +000042      06 CNAME          PIC X(5).      <-ALL-FMTS
+000043* INPUT FORMAT:ITMMNU      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000044*          ITEM INQUIRY SCREEN ONE      <-ALL-FMTS
109 +000045      05 ITMMNU-I      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
110 +000046      06 ITMMNU-I-INDIC.      <-ALL-FMTS
111 +000047          07 IN99          PIC 1 INDIC 99.      <-ALL-FMTS
112 +000048          07 IN98          PIC 1 INDIC 98.      <-ALL-FMTS
113 +000049          07 IN97          PIC 1 INDIC 97.      <-ALL-FMTS
114 +000050      06 ITEMN0          PIC S9(6).      <-ALL-FMTS
+000051* OUTPUT FORMAT:ITMMNU      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000052*          ITEM INQUIRY SCREEN ONE      <-ALL-FMTS
+000053*      05 ITMMNU-0      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
+000054* INPUT FORMAT:ITMSC2      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000055*          ITEM INQUIRY SCREEN TWO      <-ALL-FMTS
115 +000056      05 ITMSC2-I      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
116 +000057      06 ITMSC2-I-INDIC.      <-ALL-FMTS
117 +000058          07 IN99          PIC 1 INDIC 99.      <-ALL-FMTS
118 +000059          07 IN98          PIC 1 INDIC 98.      <-ALL-FMTS
119 +000060          07 IN97          PIC 1 INDIC 97.      <-ALL-FMTS
+000061* OUTPUT FORMAT:ITMSC2      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000062*          ITEM INQUIRY SCREEN TWO      <-ALL-FMTS
120 +000063      05 ITMSC2-0      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
121 +000064          06 DSC          PIC X(36).      <-ALL-FMTS
122 +000065          06 QAVAIL          PIC S9(7).      <-ALL-FMTS
123 +000066          06 QTYH          PIC S9(7).      <-ALL-FMTS
124 +000067          06 QTYO          PIC S9(7).      <-ALL-FMTS
125 +000068          06 QTYB          PIC S9(7).      <-ALL-FMTS
126 +000069          06 UNT          PIC X(2).      <-ALL-FMTS
127 +000070          06 PR1          PIC S9(5)V9(2).      <-ALL-FMTS
128 +000071          06 PR5          PIC S9(7).      <-ALL-FMTS
129 +000072          06 UFR          PIC S9(3)V9(2).      <-ALL-FMTS
+000073* INPUT FORMAT:ITMSC3      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000074*          ITEM INQUIRY SCREEN 3      <-ALL-FMTS
130 +000075      05 ITMSC3-I      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
131 +000076      06 ITMSC3-I-INDIC.      <-ALL-FMTS
132 +000077          07 IN99          PIC 1 INDIC 99.      <-ALL-FMTS
133 +000078          07 IN98          PIC 1 INDIC 98.      <-ALL-FMTS
134 +000079          07 IN97          PIC 1 INDIC 97.      <-ALL-FMTS
+000080* OUTPUT FORMAT:ITMSC3      FROM FILE DSPFIL      OF LIBRARY ICFLIB
+000081*          ITEM INQUIRY SCREEN 3      <-ALL-FMTS

```

Figure 10-22 (Part 3 of 12). Source Program Example — CSDMUL (User-Defined Formats)


```

135 +000082      05 ITMSC3-0      REDEFINES DSPFIL-RECORD.      <-ALL-FMTS
136 +000083      06 SLMS          PIC S9(7)V9(2).                <-ALL-FMTS
137 +000084      06 SLSY          PIC S9(9)V9(2).                <-ALL-FMTS
138 +000085      06 CSTM          PIC S9(7)V9(2).                <-ALL-FMTS
139 +000086      06 CSTY          PIC S9(9)V9(2).                <-ALL-FMTS
140 +000087      06 PROFIT       PIC S9(3)V9(2).                <-ALL-FMTS
141 +000088      06 LOSTS          PIC S9(7)V9(2).                <-ALL-FMTS
+000089* INPUT FORMAT:TIMOUT FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000090* TIME OUT SCREEN <-ALL-FMTS
142 +000091      05 TIMOUT-I REDEFINES DSPFIL-RECORD. <-ALL-FMTS
143 +000092      06 TIMOUT-I-INDIC. <-ALL-FMTS
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 6
STMT SEQNBR -A 1 B...+....2....+....3....+....4....+....5....+....6....+....7...IDENTFCN S COPYNAME CHG DATE
144 +000093      07 IN99          PIC 1 INDIC 99.                <-ALL-FMTS
145 +000094      07 IN98          PIC 1 INDIC 98.                <-ALL-FMTS
146 +000095      07 IN97          PIC 1 INDIC 97.                <-ALL-FMTS
147 +000096      06 TIMRSP       PIC X(1).                    <-ALL-FMTS
+000097* OUTPUT FORMAT:TIMOUT FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000098* TIME OUT SCREEN <-ALL-FMTS
+000099*      05 TIMOUT-0 REDEFINES DSPFIL-RECORD. <-ALL-FMTS
148 005700 FD QPRINT 09/30/87
149 005800 LABEL RECORDS ARE OMITTED. 09/30/87
150 005900 01 PRINTREC. 01/14/88
151 006000 05 RC PIC 9999. 01/15/88
152 006100 05 ERRMSG PIC X(128). 01/14/88
153 006200 WORKING-STORAGE SECTION. 09/30/87
154 006300 77 STATUS-IND PIC X(2). 09/30/87
155 006400 77 STATUS-DSP PIC X(2). 09/30/87
156 006500 77 MAJ-MIN-SAV PIC X(4). 09/30/87
157 006600 77 EOF-PFILE-SW PIC X VALUE "0". 09/30/87
158 006700 77 ERR-SW PIC X VALUE "0". 09/30/87
159 006800 77 INDON PIC 1 VALUE B"1". 09/30/87
160 006900 77 INDOFF PIC 1 VALUE B"0". 09/30/87
161 007000 77 OPEN-COUNT PIC 9(1) VALUE 0. 09/30/87
162 007100 77 LEN PIC 9(10)V9(5) COMP. 09/30/87
163 007200 77 PROFM PIC 9(7)V9(2) COMP-4. 09/30/87
164 007300 77 CMD2 PIC X(31) 09/30/87
165 007400 VALUE "CPYF HEXDUMP *LIST PRTFMT(*HEX)". 09/30/87
166 007500 01 SUBKEY-VALUE. 09/30/87
167 007600 05 SUBKEY PIC 9(3) VALUE 0. 09/30/87
168 007700 01 TR-CTL-AREA. 09/30/87
169 007800 05 FILLER PIC X(2). 09/30/87
170 007900 05 PGM-DEV-NME PIC X(10). 09/30/87
171 008000 05 RCD-FMT-NME PIC X(10). 09/30/87
172 008100 01 CMNF-INDIC-AREA. 09/30/87
173 008200 05 IN90 PIC 1 INDIC 90. 09/30/87
174 008300 88 IN90-ON VALUE B"1". 09/30/87
175 008400 88 IN90-OFF VALUE B"0". 09/30/87
176 008500 01 DSPF-INDIC-AREA. 09/30/87
177 008600 05 IN23 PIC 1 INDIC 23. 09/30/87
178 008700 88 IN23-ON VALUE B"1". 09/30/87
179 008800 88 IN23-OFF VALUE B"0". 09/30/87
180 008900 05 IN97 PIC 1 INDIC 97. 09/30/87
181 009000 88 IN97-ON VALUE B"1". 09/30/87
182 009100 88 IN97-OFF VALUE B"0". 09/30/87
183 009200 05 IN98 PIC 1 INDIC 98. 09/30/87
184 009300 88 IN98-ON VALUE B"1". 09/30/87
185 009400 88 IN98-OFF VALUE B"0". 09/30/87
186 009500 05 IN99 PIC 1 INDIC 99. 09/30/87
187 009600 88 IN99-ON VALUE B"1". 09/30/87
188 009700 88 IN99-OFF VALUE B"0". 09/30/87
189 009800 01 MAJ-MIN. 09/30/87
190 009900 05 MAJ PIC X(2). 09/30/87
191 010000 05 MIN PIC X(2). 09/30/87
192 010100 01 DISPLAY-FEEDBACK. 09/30/87
193 010200 05 CMD-KEY PIC X(2). 09/30/87
194 010300 05 FILLER PIC X(10). 09/30/87
195 010400 05 RCD-FMT PIC X(10). 09/30/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 7
STMT SEQNBR -A 1 B...+....2....+....3....+....4....+....5....+....6....+....7...IDENTFCN S COPYNAME CHG DATE
010500/ 09/30/87
196 010600 PROCEDURE DIVISION. 09/30/87
010700 DECLARATIVES. 09/30/87
010800 2 10/14/87
010900***** 02/21/89
011000* * 02/21/89
011100* AN ERROR ON THE DISPLAY FILE - DSPFIL - MAKES IT INACTIVE * 02/21/89
011200* THE JOB IS ENDED. * 02/21/89
011300* * 02/21/89
011400***** 02/21/89

```

Figure 10-22 (Part 4 of 12). Source Program Example — CSDMUL (User-Defined Formats)

```

011500 DSP-ERROR SECTION. 10/05/87
011600 USE AFTER STANDARD ERROR PROCEDURE ON DSPFIL. 10/05/87
011700* 10/05/87
011800 DSPFIL-EXCEPTION. 10/05/87
197 011900 MOVE "DISPLAY ERROR. JOB TERMINATED" TO ERRMSG. 02/21/89
198 012000 WRITE PRINTREC. 02/21/89
199 012100 CLOSE CMNFIL DSPFIL QPRINT. 02/21/89
200 012200 STOP RUN. 02/21/89
012300* 10/13/87
012400***** 02/21/89
012500* * 02/21/89
012600* THIS SECTION HANDLES ERRORS ON THE CMNFIL. A PERMANENT * 02/21/89
012700* SESSION ERROR WILL END THE JOB. * 02/21/89
012800* * 02/21/89
012900***** 02/21/89
013000 CMN-ERROR SECTION. 10/14/87
013100 USE AFTER STANDARD ERROR PROCEDURE ON CMNFIL. 09/30/87
013200 CMNFIL-EXCEPTION. 09/30/87
013300***** 02/21/89
013400* CHECK THE MAJOR/MINOR CODES AND TAKE APPROPRIATE ACTION * 02/21/89
013500* MAJOR CODE 34 - INPUT EXCEPTION. * 02/21/89
013600***** 02/21/89
201 013700 IF MAJ-MIN = "3431" 09/30/87
013800* DATA TRUNCATED IN INPUT AREA. SAVE RETURN CODE. 02/22/89
202 013900 MOVE MAJ-MIN TO MAJ-MIN-SAV 09/30/87
203 014000 GO TO EXIT-DECLARATIVES. 10/13/87
014100* RECOVERABLE SESSION ERROR. CLOSE ICF FILE. 10/03/90
204 014200 IF MAJ = "83" 09/30/87
205 014300 MOVE MAJ-MIN TO RC 01/14/88
206 014400 MOVE "PROGRAM STARTED AGAIN DUE TO SESSION ERROR" 09/30/87
014500 TO ERRMSG 01/14/88
207 014600 WRITE PRINTREC 09/30/87
208 014700 MOVE "1" TO ERR-SW 09/30/87
209 014800 GO TO EXIT-DECLARATIVES. 09/30/87
014900* 09/30/87
015000***** 02/21/89
015100* WHEN THERE IS A PERMANENT SESSION ERROR DETECTED, * 02/21/89
015200* THE MAJOR-MINOR CODE IS PLACED INTO A DATABASE * 02/21/89
015300* FILE AND THE FILE IS PRINTED IN HEX USING COPYFILE. * 02/21/89
015400***** 02/21/89
015500* 09/30/87
015600 GETFBA. 09/30/87
210 015700 MOVE MAJ-MIN TO RC. 01/14/88
211 015800 MOVE "PROGRAM TERMINATED DUE TO ERROR IN CMNFIL FILE" 09/30/87
015900 TO ERRMSG. 01/14/88
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 8
STMT SEQNBR -A 1 B..+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
212 016000 WRITE PRINTREC. 09/30/87
213 016100 CLOSE CMNFIL DSPFIL QPRINT. 09/30/87
214 016200 STOP RUN. 09/30/87
016300* 10/02/87
016400 EXIT-DECLARATIVES. 09/30/87
016500 EXIT. 02/28/89
016600* 09/30/87
215 016700 END DECLARATIVES. 09/30/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 9
STMT SEQNBR -A 1 B..+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
016800/ 09/30/87
016900 START-PROGRAM SECTION. 09/30/87
017000* 09/30/87
017100 START-PROGRAM-PARAGRAPH. 09/30/87
017200* 3 09/30/87
216 017300 OPEN I-O CMNFIL DSPFIL 09/30/87
017400 OUTPUT QPRINT. 09/30/87
217 017500 MOVE ZEROS TO CMNF-INDIC-AREA. 09/30/87
017600* 09/30/87
017700***** 09/30/87
017800* THE FOLLOWING TEST IS TO ATTEMPT RECOVERY IF AN ERROR * 09/30/87
017900* OCCURS WHEN OPENING THE ICF FILE. * 10/03/90
018000***** 09/30/87
018100* 4 09/30/87
218 018200 IF ERR-SW = "1" 09/30/87
219 018300 THEN IF OPEN-COUNT IS = 9 09/30/87
220 018400 THEN PERFORM DETACH-ROUTINE THRU DETACH-EXIT 09/30/87
221 018500 GO TO END-JOB 09/30/87
018600 ELSE 09/30/87
222 018700 ADD 1 TO OPEN-COUNT 09/30/87
223 018800 PERFORM ERROR-RECOVERY 09/30/87
224 018900 GO TO START-PROGRAM-PARAGRAPH 09/30/87
019000 ELSE 09/30/87

```

Figure 10-22 (Part 5 of 12). Source Program Example — CSDMUL (User-Defined Formats)

```

225 019100      MOVE 0 TO OPEN-COUNT.                                09/30/87
019200*                                               09/30/87
019300*****                                               09/30/87
019400*                                               09/30/87
019500* THE DISPLAY DEVICE IS IMPLICITLY ACQUIRED WHEN THE      *      10/15/87
019600* FILE IS OPENED.                                         *      09/30/87
019700*                                               *      09/30/87
019800* ALL OF THE ICF PROGRAM DEVICES ARE EXPLICITLY ACQUIRED. *      10/03/90
019900*                                               *      09/30/87
020000* EACH OF THE FOUR TARGET PROGRAMS ARE EVOKED TO ESTABLISH *      09/30/87
020100* TRANSACTIONS WITH THE REMOTE SYSTEMS.                   *      09/30/87
020200*                                               *      09/30/87
020300* THE MAIN INQUIRY MENU (CIMENU) IS WRITTEN TO THE USER'S *      09/30/87
020400* DISPLAY.                                                 *      09/30/87
020500*                                               *      09/30/87
020600* EVOKE PROGRAM "CTDMUL" ON REMOTE SYSTEM IN LIBRARY ICFLIB.*      10/13/87
020700*****                                               09/30/87
020800* 5                                               09/30/87
226 020900      ACQUIRE "ICF00 " FOR CMNFIL.                      09/30/87
227 021000      ACQUIRE "ICF01 " FOR CMNFIL.                      09/30/87
228 021100      ACQUIRE "ICF02 " FOR CMNFIL.                      09/30/87
229 021200      ACQUIRE "ICF03 " FOR CMNFIL.                      09/30/87
230 021300      PERFORM EVOKE-ROUTINE THRU EVOKE-EXIT.            09/30/87
021400*                                               09/30/87
231 021500      WRITE DSPREC FORMAT IS "CIMENU"                    09/30/87
021600      INDICATORS ARE DSPF-INDIC-AREA.                       09/30/87
021700*                                               10/14/87
021800*****                                               09/30/87
021900*                                               *      09/30/87
022000* DETERMINE USER'S REQUEST                                 *      09/30/87
022100*                                               *      09/30/87
022200* A READ TO THE DISPLAY DEVICE IS ISSUED TO RECEIVE      *      10/15/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 10
STMT SEQNBR -A 1 B..+...2...+...3...+...4...+...5...+...6...+...7...IDENTFCN S COPYNAME CHG DATE
022300* THE USER'S REQUEST. THE TYPE OF REQUEST MADE IS BASED ON THE *      10/13/87
022400* DISPLAY FORMAT CURRENTLY ON THE SCREEN. THE RECORD FORMAT *      10/13/87
022500* NAME IS EXTRACTED FROM THE I/O FEEDBACK AREA FOR THE DISPLAY *      10/13/87
022600* FILE AND USED TO DETERMINE WHAT ACTION SHOULD BE TAKEN NEXT. *      10/13/87
022700*                                               *      09/30/87
022800*****                                               09/30/87
022900* 6                                               09/30/87
023000 READRQ.                                               09/30/87
232 023100      READ DSPFIL INDICATORS ARE DSPF-INDIC-AREA.      09/30/87
233 023200      IF RCD-FMT = "CIMENU"                             09/30/87
234 023300      PERFORM MENU-ROUTINE THRU MENU-EXIT              09/30/87
235 023400      GO TO READRQ.                                     09/30/87
236 023500      IF RCD-FMT = "ITMNU"                             09/30/87
237 023600      PERFORM ITMIN-ROUTINE THRU ITMIN-EXIT            09/30/87
238 023700      GO TO READRQ.                                     09/30/87
239 023800      IF RCD-FMT = "ITMSC2"                            09/30/87
240 023900      PERFORM ITMRTN-ROUTINE THRU ITMRTN-EXIT          09/30/87
241 024000      GO TO READRQ.                                     09/30/87
242 024100      IF RCD-FMT = "ITMSC3"                            09/30/87
243 024200      PERFORM ITMRTN-ROUTINE THRU ITMRTN-EXIT          09/30/87
244 024300      GO TO READRQ.                                     09/30/87
245 024400      IF RCD-FMT = "DTLMNU"                             09/30/87
246 024500      PERFORM DTLIN-ROUTINE THRU DTLIN-EXIT            09/30/87
247 024600      GO TO READRQ.                                     09/30/87
248 024700      IF RCD-FMT = "DTLSCR"                             10/12/87
249 024800      PERFORM DTLRTN-ROUTINE THRU DTLRTN-EXIT          10/12/87
250 024900      GO TO READRQ.                                     10/12/87
251 025000      WRITE DSPREC FORMAT IS "CIMENU".                  09/30/87
025100                                               10/12/87
252 025200      GO TO READRQ.                                     09/30/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 11
STMT SEQNBR -A 1 B..+...2...+...3...+...4...+...5...+...6...+...7...IDENTFCN S COPYNAME CHG DATE
025300/                                               09/30/87
025400*****                                               09/30/87
025500*                                               *      09/30/87
025600* MAIN MENU                                               *      09/30/87
025700*                                               *      09/30/87
025800* THE MAIN MENU IS READ TO DETERMINE THE REQUEST ENTERED   *      10/12/87
025900* BY THE USER. IF CMD 1 (*IN99) IS PRESSED, THE PROGRAM   *      10/12/87
026000* IS ENDED. IF OPTION = 1, AN ITEM INQUIRY MENU IS WRITTEN TO *      10/12/87
026100* TO SCREEN. IF OPTION = 2, A CUSTOMER INQUIRY MENU IS    *      10/12/87
026200* WRITTEN TO THE SCREEN.                                   *      10/12/87
026300*                                               *      09/30/87
026400*****                                               09/30/87
026500* 7                                               09/30/87
026600 MENU-ROUTINE.                                           09/30/87

```

Figure 10-22 (Part 6 of 12). Source Program Example — CSDMUL (User-Defined Formats)

```

253 026700 IF CMD-KEY = "01" 09/30/87
254 026800 PERFORM DETACH-ROUTINE THRU DETACH-EXIT 09/30/87
255 026900 GO TO END-JOB. 09/30/87
256 027000 IF OPTION = "1" 09/30/87
257 027100 WRITE DSPREC FORMAT IS "ITMMNU" 09/30/87
027200 ELSE 09/30/87
258 027300 WRITE DSPREC FORMAT IS "DTLMNU". 09/30/87
027400 MENU-EXIT. 09/30/87
027500 EXIT. 09/30/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 12
STMT SEQNBR -A 1 B.....2.....3.....4.....5.....6.....7..IDENTFCN S COPYNAME CHG DATE
027600/ 09/30/87
027700***** 09/30/87
027800* * 09/30/87
027900* ITEM INQUIRY * 09/30/87
028000* * 09/30/87
028100* THE ITEM NUMBER REQUESTED BY THE USER ON THE ITEM INQUIRY * 09/30/87
028200* SCREEN IS CHECKED. THIS IS DETERMINED BY THE * 09/30/87
028300* DISPLAY RECORD FORMAT BEING PROCESSED - IN THIS CASE ITMMNU. * 09/30/87
028400* * 02/21/89
028500* IF CMD KEY 1 IS PRESSED, THE PROGRAM IS ENDED. IF CMD KEY 2 * 10/13/87
028600* IS PRESSED, THE ITEM INQUIRY REQUEST IS CANCELED, AND THE * 09/30/87
028700* MAIN MENU (CIMENU) IS WRITTEN TO THE SCREEN. * 09/30/87
028800* * 09/30/87
028900* IF AN ITEM NUMBER IS ENTERED, AN ITEM INQUIRY REQUEST IS * 09/30/87
029000* SENT TO THE APPROPRIATE REMOTE SYSTEM. THE REMOTE SYSTEM * 09/30/87
029100* IS SELECTED BASED ON THE ITEM NUMBER REQUESTED. * 09/30/87
029200* * 09/30/87
029300* A CHECK IS MADE FOR THREE CONDITIONS FOLLOWING THE READ. * 10/14/87
029400* 1) THE REMOTE SYSTEM TIMED OUT, 2) NO DATA RECEIVED, AND * 10/14/87
029500* 3) DATA RETURNED IN AN UNEXPECTED RECORD FORMAT. * 10/14/87
029600* * 10/14/87
029700* IF THE REMOTE SYSTEM TIMES OUT (MAJ-MIN = 0310) A MESSAGE * 10/14/87
029800* IS WRITTEN TO THE SCREEN, ASKING TO TRY AGAIN OR END THE * 10/14/87
029900* PROGRAM. * 10/14/87
030000* * 10/14/87
030100* IF NO DATA IS RECEIVED AFTER THE READ OPERATION TO THE * 10/15/87
030200* PROGRAM DEVICE (MAJ-MIN = 03__ ) THE REQUEST IS SENT AGAIN * 10/14/87
030300* TO THE REMOTE SYSTEM AND THE READ OPERATION IS ISSUED TO * 10/14/87
030400* THE PROGRAM DEVICE. * 10/15/87
030500* * 10/14/87
030600* IF THE RECORD RETURNS WITH THE WRONG RECORD FORMAT, THE * 10/14/87
030700* PROGRAM WILL GO TO EXIT-FORMAT-ERR ROUTINE. * 10/14/87
030800* * 10/14/87
030900***** 09/30/87
031000* 8 09/30/87
259 031100 ITMIN-ROUTINE. 09/30/87
260 031200 IF CMD-KEY = "01" 09/30/87
261 031300 PERFORM DETACH-ROUTINE THRU DETACH-EXIT 10/12/87
262 031400 GO TO END-JOB. 10/12/87
263 031500 IF CMD-KEY = "02" 10/12/87
264 031600 WRITE DSPREC FORMAT IS "CIMENU" 10/12/87
265 031700 GO TO ITMIN-EXIT. 10/12/87
266 031800 MOVE CORR ITMMNU-I TO ITMREQ. 09/30/87
* ** CORRESPONDING items for statement 266:
* ** ITEMNO
* ** End of CORRESPONDING items for statement 266
267 031900 IF ITEMNO OF ITMMNU-I LESS THAN 399999 GO TO XICF01. 09/30/87
269 032000 IF ITEMNO OF ITMMNU-I LESS THAN 699999 GO TO XICF02. 09/30/87
271 032100 IF ITEMNO OF ITMMNU-I LESS THAN 899999 GO TO XICF03. 09/30/87
032200 XICF01. 09/30/87
273 032300 MOVE "ICF01 " TO PGM-DEV-NME. 09/30/87
274 032400 GO TO XITMIN. 09/30/87
032500 XICF02. 09/30/87
275 032600 MOVE "ICF02 " TO PGM-DEV-NME. 09/30/87
276 032700 GO TO XITMIN. 09/30/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 13
STMT SEQNBR -A 1 B.....2.....3.....4.....5.....6.....7..IDENTFCN S COPYNAME CHG DATE
032800 XICF03. 09/30/87
277 032900 MOVE "ICF03 " TO PGM-DEV-NME. 09/30/87
033000 XITMIN. 09/30/87
278 033100 MOVE ZEROS TO CMNF-INDIC-AREA. 09/30/87
279 033200 WRITE CMNREC FORMAT IS "ITMREQ" 09/30/87
033300 TERMINAL IS PGM-DEV-NME. 09/30/87
033400 TRY-AGAIN. 10/01/87
280 033500 READ CMNFIL. 09/30/87
281 033600 IF MAJ-MIN = "0310" 10/01/87
282 033700 WRITE DSPREC FORMAT IS "TIMOUT" 09/30/87
283 033800 READ DSPFIL INDICATORS ARE DSPF-INDIC-AREA 09/30/87
284 033900 IF TIMRSP = "1" GO TO TRY-AGAIN END-IF 01/21/88

```

Figure 10-22 (Part 7 of 12). Source Program Example — CSDMUL (User-Defined Formats)

```

286 034000      IF TIMRSP = "2" GO TO END-JOB END-IF.                                01/21/88
288 034100      IF MAJ = "03"                                                        09/30/87
289 034200      GO TO XITMIN.                                                         09/30/87
290 034300      IF RCD-FMT-NME IS NOT EQUAL "ITMRSP" GO TO EXIT-FORMAT-ERR.         10/02/87
292 034400      PERFORM ITMOUT-ROUTINE THRU ITMOUT-EXIT.                             09/30/87
034500 ITMIN-EXIT.                                                                    09/30/87
034600      EXIT.                                                                    09/30/87
5738CB1 V2R1M0 910524      AS/400 COBOL Source      ICFLIB/CSDMUL      RCHAS400 10/03/90 14:33:13      Page 14
STMT SEQNBR -A 1 B..+...2....+...3....+...4....+...5....+...6....+...7..IDENTFCN S COPYNAME  CHG DATE
034700/
034800*****
034900* * * * *
035000*      PROCESS ITEM INFORMATION * * * * *
035100* * * * *
035200*      THE ITEM RECORD RECEIVED FROM THE TARGET PROGRAM AND THE * * * * *
035300*      INFORMATION ABOUT THE ITEM IS PROCESSED AND DISPLAYED. * * * * *
035400*      IF ITEMNO IS 0 OR LESS, IT IS AN INVALID REQUEST AND A FRESH * * * * *
035500*      ITEM MENU IS WRITTEN TO THE SCREEN. IF THE REQUEST IS * * * * *
035600*      VALID, VALUES ARE CALCULATED BASED ON THE INFORMATION * * * * *
035700*      RECEIVED. * * * * *
035800* * * * *
035900*****
036000* 9
293 036100      ITMOUT-ROUTINE.
294 036200      IF ITEMNO OF ITMRSP NOT GREATER THAN 0
295 036300      WRITE DSPREC FORMAT IS "ITMMNU"
296 036400      GO TO ITMOUT-EXIT.
297 036500      MOVE DESC TO DSC OF ITMSC2-0.
298 036600      MOVE QTYLST TO QAVAIL OF ITMSC2-0.
299 036700      MOVE QTY00 TO QTY0 OF ITMSC2-0.
300 036800      MOVE QTY0H TO QTYH OF ITMSC2-0.
301 036900      MOVE QTYB0 TO QTYB OF ITMSC2-0.
302 037000      MOVE UNITQ TO UNT OF ITMSC2-0.
303 037100      MOVE PR01 TO PR1 OF ITMSC2-0.
304 037200      MOVE PR05 TO PR5 OF ITMSC2-0.
305 037300      MOVE UFRT TO UFR OF ITMSC2-0.
306 037400      WRITE DSPREC FORMAT IS "ITMSC2"
037500      INDICATORS ARE DSPF-INDIC-AREA.
037600 ITMOUT-EXIT.
037700      EXIT.
037800* * * * *
037900*****
038000* * * * *
038100*      ADDITIONAL ITEM INFORMATION * * * * *
038200* * * * *
038300*      ADDITIONAL ITEM INFORMATION IS PROCESSED AND THE RESULT * * * * *
038400*      DISPLAYED ON THE SCREEN WHEN A RESPONSE IS READ FROM THE * * * * *
038500*      DISPLAY STATION WITH AN ITEM SCREEN RECORD FORMAT. * * * * *
038600* * * * *
038700*      IF CMD KEY 1 IS PRESSED, THE PROGRAM IS ENDED. IF CMD KEY 2 * * * * *
038800*      IS PRESSED, THE ITEM INQUIRY IS ENDED, AND THE MAIN MENU * * * * *
038900*      (CIMENU) IS WRITTEN TO THE SCREEN. IF CMD KEY 3 IS PRESSED, * * * * *
039000*      THE ITEM INQUIRY MENU IS WRITTEN TO THE SCREEN. BY PRESSING * * * * *
039100*      ENTER WHEN SCREEN 2 IS DISPLAYED, MORE INFORMATION (PROFIT- * * * * *
039200*      LOSS) IS WRITTEN TO THE SCREEN. IF SCREEN 3 IS DISPLAYED, * * * * *
039300*      PRESSING ENTER WILL CAUSE THE ITEM INQUIRY MENU TO BE * * * * *
039400*      WRITTEN TO THE SCREEN. * * * * *
039500* * * * *
039600*****
039700* 10
307 039800      ITMRTN-ROUTINE.
308 039900      IF CMD-KEY = "01"
309 040000      PERFORM DETACH-ROUTINE THRU DETACH-EXIT
310 040100      GO TO END-JOB.
5738CB1 V2R1M0 910524      AS/400 COBOL Source      ICFLIB/CSDMUL      RCHAS400 10/03/90 14:33:13      Page 15
STMT SEQNBR -A 1 B..+...2....+...3....+...4....+...5....+...6....+...7..IDENTFCN S COPYNAME  CHG DATE
311 040200      IF CMD-KEY = "02"
312 040300      WRITE DSPREC FORMAT IS "CIMENU"
313 040400      GO TO ITMRTN-EXIT.
314 040500      IF CMD-KEY = "03"
315 040600      WRITE DSPREC FORMAT IS "ITMMNU"
316 040700      GO TO ITMRTN-EXIT.
317 040800      IF RCD-FMT = "ITMSC2"
318 040900      PERFORM PROFIT-LOSS THRU PROFIT-LOSS-EXIT
319 041000      WRITE DSPREC FORMAT IS "ITMSC3"
320 041100      GO TO ITMRTN-EXIT.
321 041200      WRITE DSPREC FORMAT IS "ITMMNU".
041300 ITMRTN-EXIT.
041400      EXIT.
041500* * * * *

```

Figure 10-22 (Part 8 of 12). Source Program Example — CSDMUL (User-Defined Formats)

```

041600*****
041700*
041800*   PROFIT AND LOSS FIGURES ARE CALCULATED FOR THE ITEM NUMBER
041900*   REQUESTED.  THESE ARE USED IN SCREEN 2 OF THE ITEM.
042000*
042100*****
042200*
322 042300 PROFIT-LOSS.
042400* 11
323 042500   SUBTRACT SLSTM FROM CSTTM GIVING PROFM.
324 042600   MULTIPLY PROFM BY 100 GIVING PROFM.
325 042700   IF SLSTM GREATER THAN 0
326 042800       DIVIDE PROFM BY SLSTM GIVING PROFM.
327 042900   MULTIPLY QTYLST BY PR01 GIVING LOSTS.
328 043000   MOVE SLSTM TO SLSM.
329 043100   MOVE SLSTY TO SLSY.
330 043200   MOVE CSTTM TO CSTM.
331 043300   MOVE PROFM TO PROFIT.
332 043400   MOVE CSTTY TO CSTY.
043500 PROFIT-LOSS-EXIT.
043600   EXIT.
5738CB1 V2R1M0 910524          AS/400 COBOL Source          ICLIB/CSDMUL          RCHAS400 10/03/90 14:33:13      Page 16
STMT SEQNBR -A 1 B.....2.....3.....4.....5.....6.....7..IDENTFCN S COPYNAME  CHG DATE
043700/
043800*****
043900*
044000*   CUSTOMER INQUIRY
044100*
044200*   THE REQUEST FROM THE CUSTOMER INQUIRY MENU IS PROCESSED.
044300*   IF CMD KEY 1 IS PRESSED, THE PROGRAM IS ENDED.  IF CMD KEY 2
044400*   IS PRESSED, THE MAIN MENU (CIMENU) IS WRITTEN TO THE SCREEN.
044500*
044600*   IF A CUSTOMER NUMBER IS ENTERED, THE CUSTOMER INQUIRY
044700*   REQUEST IS SENT TO THE REMOTE SYSTEM.  THEN DTOUT-ROUTINE
044800*   THRU DTOUT-EXIT ARE PERFORMED.
044900*
045000*****
045100* 12
333 045200 DTLIN-ROUTINE.
334 045300   IF CMD-KEY = "01"
335 045400       PERFORM DETACH-ROUTINE THRU DETACH-EXIT
336 045500       GO TO END-JOB.
337 045600   IF CMD-KEY = "02"
338 045700       WRITE DSPREC FORMAT IS "CIMENU"
339 045800       GO TO DTLIN-EXIT.
045900 EVDTL.
340 046000   MOVE "ICF00 " TO PGM-DEV-NME.
341 046100   MOVE CORR DTLMNU-I TO DTLREQ.
*
*   ** CORRESPONDING items for statement 341:
*   ** CUSTNO
*   ** End of CORRESPONDING items for statement 341
342 046200   MOVE ZEROS TO CMNF-INDIC-AREA.
343 046300   WRITE CMNREC FORMAT IS "DTLREQ"
046400   TERMINAL IS PGM-DEV-NME.
344 046500   PERFORM DTOUT-ROUTINE THRU DTOUT-EXIT.
046600 DTLIN-EXIT.
046700   EXIT.
046800*
046900*****
047000*
047100*   PROCESS CUSTOMER INFORMATION
047200*
047300*   THE CUSTOMER DATA RECEIVED FROM THE TARGET PROGRAM IS
047400*   PROCESSED.  IF CUSTOMER NUMBER IS ZERO OR LESS, IT IS AN
047500*   INVALID REQUEST AND THE MAIN MENU IS WRITTEN TO THE SCREEN.
047600*
047700*****
047800* 13
345 047900 DTOUT-ROUTINE.
346 048000   IF CUSTNO OF DTLRSP-I NOT GREATER THAN 0
347 048100       WRITE DSPREC FORMAT IS "CIMENU"
348 048200       GO TO DTOUT-EXIT.
349 048300   PERFORM CUSTOMER-DETAIL THRU CUSTOMER-DETAIL-EXIT.
048400 DTOUT-EXIT.
048500   EXIT.
048600*
048700*****
048800*

```

Figure 10-22 (Part 9 of 12). Source Program Example — CSDMUL (User-Defined Formats)

```

5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 17
STMT SEQNBR -A 1 B.+. . . . 2. . . . 3. . . . 4. . . . 5. . . . 6. . . . 7. . . . IDENTFCN S COPYNAME CHG DATE
048900* THIS ROUTINE HANDLES THE USER'S REQUEST FOLLOWING THE DISPLAY * 02/22/89
049000* OF THE CUSTOMER INFORMATION. CMD KEY 1 WILL EXIT THE JOB, * 02/22/89
049100* CMD KEY 2 WILL DISPLAY THE MAIN MENU, AND "ENTER" WILL BRING * 02/22/89
049200* UP THE CUSTOMER INQUIRY MENU. * 02/22/89
049300* * 02/22/89
049400***** 02/22/89
049500* 14 10/14/87
350 049600 DTLRTN-ROUTINE. 10/12/87
351 049700 IF CMD-KEY = "01" 10/12/87
352 049800 PERFORM DETACH-ROUTINE THRU DETACH-EXIT 10/12/87
353 049900 GO TO END-JOB. 10/12/87
354 050000 IF CMD-KEY = "02" 10/12/87
355 050100 WRITE DSPREC FORMAT IS "CIMENU" 10/12/87
356 050200 GO TO DTLRTN-EXIT. 10/12/87
357 050300 WRITE DSPREC FORMAT IS "DTLMNU". 10/12/87
050400 DTLRTN-EXIT. 10/12/87
050500 EXIT. 10/12/87
050600* 10/12/87
050700***** 02/21/89
050800* * 02/21/89
050900* THE READ OPERATION TO THE PROGRAM DEVICE IS ISSUED. * 02/21/89
051000* A CHECK IS MADE FOR THREE CONDITIONS FOLLOWING THE READ. * 02/21/89
051100* 1) THE REMOTE SYSTEM TIMED OUT, 2) NO DATA RECEIVED, AND * 02/21/89
051200* 3) DATA RETURNED IN AN UNEXPECTED RECORD FORMAT. * 02/21/89
051300* * 02/21/89
051400* IF THE REMOTE SYSTEM TIMES OUT (MAJ-MIN = 0310) A MESSAGE * 02/21/89
051500* IS WRITTEN TO THE SCREEN, ASKING TO TRY AGAIN OR END THE * 02/21/89
051600* PROGRAM. * 02/21/89
051700* * 02/21/89
051800* IF NO DATA IS RECEIVED AFTER THE READ OPERATION TO THE * 02/21/89
051900* PROGRAM DEVICE (MAJ-MIN = 03__ ) THE REQUEST IS SENT AGAIN * 02/21/89
052000* TO THE REMOTE SYSTEM AND THE READ OPERATION IS ISSUED TO * 02/21/89
052100* THE ICF PROGRAM DEVICE. * 10/03/90
052200* * 02/21/89
052300* IF THE RECORD RETURNS WITH THE WRONG RECORD FORMAT, THE * 02/21/89
052400* PROGRAM WILL GO TO EXIT-FORMAT-ERR ROUTINE. * 02/21/89
052500* * 02/22/89
052600***** 02/21/89
052700* 15 10/14/87
052800* 09/30/87
358 052900 CUSTOMER-DETAIL. 09/30/87
359 053000 MOVE ZEROS TO CMNF-INDIC-AREA. 09/30/87
360 053100 READ CMNFIL. 09/30/87
361 053200 IF MAJ-MIN = "0310" 10/01/87
362 053300 WRITE DSPREC FORMAT IS "TIMOUT" 09/30/87
363 053400 READ DSPFIL INDICATORS ARE DSPF-INDIC-AREA 09/30/87
364 053500 IF TIMRSP = "1" GO TO CUSTOMER-DETAIL END-IF 01/21/88
366 053600 IF TIMRSP = "2" GO TO END-JOB END-IF. 01/21/88
368 053700 IF MAJ = "03" 09/30/87
369 053800 MOVE ZEROS TO CMNF-INDIC-AREA 09/30/87
370 053900 WRITE CMNREC FORMAT IS "DTLREQ" 09/30/87
054000 TERMINAL IS PGM-DEV-NME 09/30/87
371 054100 GO TO CUSTOMER-DETAIL. 09/30/87
372 054200 IF RCD-FMT-NME IS NOT EQUAL "DTLRSP" GO TO EXIT-FORMAT-ERR. 10/02/87
374 054300 MOVE CUSTNO OF DTLRSP-I TO CUSTN OF DTLSCR-0. 10/12/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 18
STMT SEQNBR -A 1 B.+. . . . 2. . . . 3. . . . 4. . . . 5. . . . 6. . . . 7. . . . IDENTFCN S COPYNAME CHG DATE
375 054400 MOVE DNAME OF DTLRSP-I TO CNAME OF DTLSCR-0. 03/21/89
376 054500 MOVE DLSTOR OF DTLRSP-I TO DLSTR OF DTLSCR-0. 10/12/87
377 054600 MOVE DSLSTM OF DTLRSP-I TO DSLSM OF DTLSCR-0. 10/12/87
378 054700 MOVE DSPM01 OF DTLRSP-I TO DSPM1 OF DTLSCR-0. 10/12/87
379 054800 MOVE DSPM02 OF DTLRSP-I TO DSPM2 OF DTLSCR-0. 10/12/87
380 054900 MOVE DSTTYD OF DTLRSP-I TO DSTYD OF DTLSCR-0. 10/12/87
381 055000 MOVE IDEPT OF DTLRSP-I TO DEPT OF DTLSCR-0. 10/12/87
382 055100 WRITE DSPREC FORMAT IS "DTLSCR". 10/12/87
055200 CUSTOMER-DETAIL-EXIT. 09/30/87
055300 EXIT. 09/30/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 19
STMT SEQNBR -A 1 B.+. . . . 2. . . . 3. . . . 4. . . . 5. . . . 6. . . . 7. . . . IDENTFCN S COPYNAME CHG DATE
055400/ 09/30/87
055500***** 02/21/89
055600* * 02/21/89
055700* THE EVOKE-ROUTINE IS CALLED TO EVOKE THE TARGET PROGRAM. * 02/21/89
055800* THE SAME TARGET PROGRAM (ICFLIB/CTDMULCL) IS EVOKED AT * 02/21/89
055900* FOUR DIFFERENT REMOTE SYSTEMS. THE PROGRAM DEVICE * 02/21/89
056000* IDENTIFIES WHICH SESSION SHOULD BE EVOKED. THE PROGRAM * 02/21/89
056100* DEVICE WAS SPECIFIED IN CMID PRIOR TO CALLING THIS ROUTINE. * 02/21/89
056200* * 02/21/89

```

Figure 10-22 (Part 10 of 12). Source Program Example — CSDMUL (User-Defined Formats)

```

056300*****
056400* 16
056500*
383 056600 EVOKE-ROUTINE.
384 056700 MOVE "CTDMULCL" TO PGMID OF EVKREQ-O.
385 056800 MOVE "ICFLIB" TO LIB OF EVKREQ-O.
386 056900 MOVE "ICF00 " TO PGM-DEV-NME
387 057000 WRITE CMNREC FORMAT IS "EVKREQ"
057100 TERMINAL IS PGM-DEV-NME.
388 057200 MOVE "ICF01 " TO PGM-DEV-NME
389 057300 WRITE CMNREC FORMAT IS "EVKREQ"
057400 TERMINAL IS PGM-DEV-NME.
390 057500 MOVE "ICF02 " TO PGM-DEV-NME
391 057600 WRITE CMNREC FORMAT IS "EVKREQ"
057700 TERMINAL IS PGM-DEV-NME.
392 057800 MOVE "ICF03 " TO PGM-DEV-NME
393 057900 WRITE CMNREC FORMAT IS "EVKREQ"
058000 TERMINAL IS PGM-DEV-NME.
058100 EVOKE-EXIT.
058200 EXIT.
058300*
058400*****
058500* *
058600* THE TRANSACTION AND SESSION ARE ENDED FOR EACH OF THE *
058700* REMOTE SYSTEMS. *
058800* *
058900*****
059000* 17
394 059100 ERROR-RECOVERY.
395 059200 PERFORM DETACH-ROUTINE THRU DETACH-EXIT.
396 059300 CLOSE CMNFIL DSPFIL
059400 QPRINT.
397 059500 MOVE "0" TO ERR-SW.
059600 ERROR-RECOVERY-EXIT.
059700 EXIT.
059800*****
059900* *
060000* EXIT-FORMAT-ERR IS PERFORMED WHEN A READ TO CMNFIL RETURNS WITH *
060100* AN UNEXPECTED RCD-FMT-NME IN THE I-O-FEEDBACK AREA FOR CMNFIL. *
060200* AN ERROR MESSAGE IS PRINTED AND THE PROGRAM ENDS. *
060300* *
060400*****
060500* 18
398 060600 EXIT-FORMAT-ERR.
399 060700 MOVE MAJ-MIN TO RC.
400 060800 MOVE "RECORD FORMAT IS INCORRECT ON READ "
5738CB1 V2RIM0 910524 AS/400 COBOL Source ICFLIB/CSDMUL RCHAS400 10/63/90 14:33:13 Page 20
STMT SEQNBR -A 1 B...+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
060900 TO ERRMSG.
061000 WRITE PRINTREC.
061100 CLOSE CMNFIL DSPFIL QPRINT.
061200 STOP RUN.
061300*
061400*****
061500* *
061600* THIS ROUTINE IS CALLED TO END THE TRANSACTION WITH THE *
061700* REMOTE SYSTEM. *
061800* *
061900*****
062000* 19
062100 DETACH-ROUTINE.
404 062200 MOVE "ICF00 " TO PGM-DEV-NME
405 062300 WRITE CMNREC FORMAT IS "DETACH"
062400 TERMINAL IS PGM-DEV-NME.
406 062500 MOVE "ICF01 " TO PGM-DEV-NME
407 062600 WRITE CMNREC FORMAT IS "DETACH"
062700 TERMINAL IS PGM-DEV-NME.
408 062800 MOVE "ICF02 " TO PGM-DEV-NME
409 062900 WRITE CMNREC FORMAT IS "DETACH"
063000 TERMINAL IS PGM-DEV-NME.
410 063100 MOVE "ICF03 " TO PGM-DEV-NME
411 063200 WRITE CMNREC FORMAT IS "DETACH"
063300 TERMINAL IS PGM-DEV-NME.
063400 DETACH-EXIT.
063500 EXIT.
063600*

```

Figure 10-22 (Part 11 of 12). Source Program Example — CSDMUL (User-Defined Formats)


```

063700*****
063800*
063900* THIS ROUTINE IS CALLED TO RELEASE THE PROGRAM DEVICES, END *
064000* THE SESSION AND END THE PROGRAM. *
064100* *
064200*****
064300* 20
064400*
412 064500 END-JOB.
413 064600 DROP "ICF00 " FROM CMNFIL.
414 064700 DROP "ICF01 " FROM CMNFIL.
415 064800 DROP "ICF02 " FROM CMNFIL.
416 064900 DROP "ICF03 " FROM CMNFIL.
417 065000 CLOSE CMNFIL DSPFIL QPRINT.
418 065100 STOP RUN.
065200*
***** END OF SOURCE *****
5738CB1 V2R1M0 910524 AS/400 COBOL Messages ICFLIB/CSDMUL RCHAS400 10/03/90 14:33:13 Page 21
STMT
* 25 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 005200
Message . . . . : No INPUT fields found for format DETACH.
* 25 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 005200
Message . . . . : No INPUT fields found for format EOS.
* 25 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 005200
Message . . . . : No INPUT fields found for format EVKREQ.
* 80 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 005600
Message . . . . : No OUTPUT fields found for format CIMENU.
* 80 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 005600
Message . . . . : No OUTPUT fields found for format DTLMNU.
* 80 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 005600
Message . . . . : No OUTPUT fields found for format ITMMNU.
* 80 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 005600
Message . . . . : No OUTPUT fields found for format TIMOUT.
***** END OF MESSAGES *****
Message Summary
Total Info(0-4) Warning(5-19) Error(20-29) Severe(30-39) Terminal(40-99)
7 0 7 0 0 0
Source records read . . . . . : 652
Copy records read . . . . . : 171
Copy members processed . . . . . : 2
Sequence errors . . . . . : 0
Highest severity message issued . . : 10
LBL0901 00 Program CSDMUL created in library ICFLIB.
***** END OF COMPILATION *****

```

Figure 10-22 (Part 12 of 12). Source Program Example — CSDMUL (User-Defined Formats)

```

5738CB1 V2R1M0 910524          IBM AS/400 COBOL/400          ICFLIB/CSFMUL          RCHAS400 10/03/90 14:37:21          Page 1
Program . . . . . : CSFMUL
Library . . . . . : ICFLIB
Source file . . . . . : QICFPUB
Library . . . . . : ICFLIB
Source member . . . . . : CSFMUL 10/03/90 14:26:27
Generation severity level . . . . . : 29
Text 'description' . . . . . : CBL Multiple Session Inquiry - Source $$
Source listing options . . . . . : *SOURCE
Generation options . . . . . : *NONE
Message limit:
Number of messages . . . . . : *NOMAX
Message limit severity . . . . . : 29
Print file . . . . . : QSYSVRT
Library . . . . . : *LIBL
FIPS flagging . . . . . : *NOFIPS *NOSEG *NODEB *NOBSOLETE
SAA flagging . . . . . : *NOFLAG
Flagging severity . . . . . : 0
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Compiler . . . . . : IBM AS/400 COBOL/400

```

```

5738CB1 V2R1M0 910524          AS/400 COBOL Source          ICFLIB/CSFMUL          RCHAS400 10/03/90 14:37:21          Page 2
STMT SEQNBR -A 1 B...2...3...4...5...6...7..IDENTFCN S COPYNAME CHG DATE
1 000100 IDENTIFICATION DIVISION. 10/01/87
2 000200 PROGRAM-ID. CSFMUL. 10/01/87
000300*****
000400* THIS PROGRAM ASSIGNS FOUR SESSIONS AS FOLLOWS: * 10/01/87
000500* 'ICF00' TO INQUIRE ABOUT A CUSTOMER ACCOUNT BEFORE AN * 10/01/87
000600* ORDER IS PROCESSED. * 10/01/87
000700* 'ICF01' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM * 10/01/87
000800* BEING ORDERED (ITEM 000001 THRU 399999). * 10/01/87
000900* 'ICF02' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM * 10/01/87
001000* BEING ORDERED (ITEM 400000 THRU 699999). * 10/01/87
001100* 'ICF03' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM * 10/01/87
001200* BEING ORDERED (ITEM 700000 THRU 999999). * 10/01/87
001300* A DISPLAY DEVICE IS USED TO ENTER THE REQUEST ( USING A * 10/01/87
001400* CUSTOMER AND AN ITEM MENU) THAT IS SENT TO THE REMOTE * 10/01/87
001500* SYSTEM. * 10/15/87
001600*****
3 001700 ENVIRONMENT DIVISION. 10/01/87
4 001800 CONFIGURATION SECTION. 10/01/87
5 001900 SOURCE-COMPUTER. IBM-AS400. 01/15/88
6 002000 OBJECT-COMPUTER. IBM-AS400. 01/15/88
7 002100 SPECIAL-NAMES. I-O-FEEDBACK IS I-O-FEEDBACK 10/01/87
8 002200 OPEN-FEEDBACK IS OPEN-FBA. 10/01/87
9 002300 INPUT-OUTPUT SECTION. 10/01/87
10 002400 FILE-CONTROL. 10/01/87
002500* 1 10/14/87
002600*****
002700* 10/01/87
002800* FILE SPECIFICATIONS * 10/01/87
002900* * 10/01/87
003000* CMNFIL : ICF FILE USED TO SEND A REQUEST TO ONE * 10/03/90
003100* OF FOUR DIFFERENT TARGET PROGRAMS. MULTIPLE * 10/01/87
003200* SESSIONS ARE ACTIVE CONCURRENTLY. * 10/01/87
003300* * 10/01/87
003400* DSPFIL : DISPLAY FILE USED TO ENTER A REQUEST TO BE * 10/14/87
003500* SENT TO A REMOTE SYSTEM. * 10/01/87
003600* * 10/01/87
003700*****
11 003800 SELECT CMNFIL ASSIGN TO WORKSTATION-CMNFIL-S1 10/08/87
12 003900 ORGANIZATION IS TRANSACTION 10/01/87
13 004000 CONTROL-AREA IS TR-CTL-AREA 10/01/87
14 004100 FILE STATUS IS STATUS-IND MAJ-MIN. 10/01/87
15 004200 SELECT DSPFIL ASSIGN TO WORKSTATION-DSPFIL 10/08/87
16 004300 ORGANIZATION IS TRANSACTION 10/01/87
17 004400 CONTROL-AREA IS DISPLAY-FEEDBACK 10/01/87
18 004500 FILE STATUS IS STATUS-DSP. 10/01/87
19 004600 SELECT QPRINT ASSIGN TO PRINTER-QSYSVRT. 10/01/87
20 004700 DATA DIVISION. 10/01/87
21 004800 FILE SECTION. 10/01/87
22 004900 FD CMNFIL 10/08/87
23 005000 LABEL RECORDS ARE STANDARD. 10/01/87
24 005100 01 CMNREC. 10/01/87
25 005200 05 ITMRSP. 10/01/87
26 005300 07 RECITM PIC X. 10/01/87
27 005400 07 ITEMNO PIC 9(6). 10/01/87
28 005500 07 DESC PIC X(30). 10/01/87

```

Figure 10-23 (Part 1 of 11). Source Program Example — CSFMUL (System-Supplied Formats)

```

5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 3
STMT SEQNBR -A 1 B.+.2....3....4....5....6....7..IDENTFCN S COPYNAME CHG DATE
29 005600 07 QTYLST PIC 9(7). 10/01/87
30 005700 07 QTYOH PIC 9(7). 10/01/87
31 005800 07 QTY00 PIC 9(7). 10/01/87
32 005900 07 QTYB0 PIC 9(7). 10/01/87
33 006000 07 UNITQ PIC 99. 10/01/87
34 006100 07 PR01 PIC 9(5)V99. 10/01/87
35 006200 07 PR05 PIC 9(7). 10/01/87
36 006300 07 UFRT PIC 999V99. 10/01/87
37 006400 07 SLSTM PIC 9(7)V99. 10/01/87
38 006500 07 SLSTY PIC 9(9)V99. 10/01/87
39 006600 07 CSTTM PIC 9(7)V99. 10/01/87
40 006700 07 CSTTY PIC 9(9)V99. 10/01/87
41 006800 07 PR0 PIC 999V99. 10/01/87
42 006900 07 LOS PIC 9(7)V99. 10/01/87
43 007000 07 FILL1 PIC X(56). 10/01/87
44 007100 05 ITMREQ REDEFINES ITMRSP. 10/01/87
45 007200 07 LNPTH PIC 9(4). 02/22/89
46 007300 07 ITEMNO PIC 9(6). 10/01/87
47 007400 05 DTLREQ REDEFINES ITMRSP. 10/01/87
48 007500 07 LNPTH PIC 9(4). 02/22/89
49 007600 07 CUSTNO PIC 9(6). 10/01/87
50 007700 05 DTLRSP REDEFINES ITMRSP. 10/01/87
51 007800 07 RECCUS PIC X. 10/01/87
52 007900 07 CUSTNO PIC 9(6). 10/01/87
53 008000 07 DNAME PIC X(30). 10/01/87
54 008100 07 DLSTOR PIC 9(6). 10/01/87
55 008200 07 DSLSTM PIC 9(9). 10/01/87
56 008300 07 DSPM01 PIC 9(9). 10/01/87
57 008400 07 DSPM02 PIC 9(9). 10/01/87
58 008500 07 DSPM03 PIC 9(9). 10/01/87
59 008600 07 DSTTYD PIC 9(11). 10/01/87
60 008700 07 IDEPT PIC 999. 10/01/87
61 008800 07 FILL2 PIC X(57). 10/01/87
62 008900 05 EVKREQ REDEFINES ITMRSP. 10/01/87
63 009000 07 PGMID PIC X(8). 10/01/87
64 009100 07 FILLER PIC X(16). 10/01/87
65 009200 07 LIB PIC X(8). 10/01/87
66 009300 07 FILLER PIC X(20). 10/01/87
67 009400 07 LNPTH PIC 9(4). 02/22/89
68 009500 FD DSPFIL 10/08/87
69 009600 LABEL RECORDS ARE STANDARD. 10/01/87
70 009700 01 DSPREC. 10/01/87
71 009800 COPY DDS-ALL-FORMATS-I-0 OF DSPFIL. 10/08/87
72 +000001 05 DSPFIL-RECORD PIC X(79). <-ALL-FMTS
+000002* INPUT FORMAT:CIMENU FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000003* MENU FOR INQUIRY <-ALL-FMTS
73 +000004 05 CIMENU-I REDEFINES DSPFIL-RECORD. <-ALL-FMTS
74 +000005 06 CIMENU-I-INDIC. <-ALL-FMTS
75 +000006 07 IN99 PIC 1 INDIC 99. <-ALL-FMTS
76 +000007 07 IN98 PIC 1 INDIC 98. <-ALL-FMTS
77 +000008 07 IN97 PIC 1 INDIC 97. <-ALL-FMTS
78 +000009 06 OPTION PIC X(1). <-ALL-FMTS
+000010* OUTPUT FORMAT:CIMENU FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000011* MENU FOR INQUIRY <-ALL-FMTS
+000012* 05 CIMENU-0 REDEFINES DSPFIL-RECORD. <-ALL-FMTS
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 4
STMT SEQNBR -A 1 B.+.2....3....4....5....6....7..IDENTFCN S COPYNAME CHG DATE
+000013* INPUT FORMAT:DTLMNU FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000014* CUSTOMER INQUIRY SCREEN 1 <-ALL-FMTS
79 +000015 05 DTLMNU-I REDEFINES DSPFIL-RECORD. <-ALL-FMTS
80 +000016 06 DTLMNU-I-INDIC. <-ALL-FMTS
81 +000017 07 IN99 PIC 1 INDIC 99. <-ALL-FMTS
82 +000018 07 IN98 PIC 1 INDIC 98. <-ALL-FMTS
83 +000019 07 IN97 PIC 1 INDIC 97. <-ALL-FMTS
84 +000020 06 CUSTNO PIC S9(6). <-ALL-FMTS
+000021* OUTPUT FORMAT:DTLMNU FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000022* CUSTOMER INQUIRY SCREEN 1 <-ALL-FMTS
+000023* 05 DTLMNU-0 REDEFINES DSPFIL-RECORD. <-ALL-FMTS
+000024* INPUT FORMAT:DTLSCR FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000025* CUSTOMER INQUIRY SCR. #2 <-ALL-FMTS
85 +000026 05 DTLSCR-I REDEFINES DSPFIL-RECORD. <-ALL-FMTS
86 +000027 06 DTLSCR-I-INDIC. <-ALL-FMTS
87 +000028 07 IN99 PIC 1 INDIC 99. <-ALL-FMTS
88 +000029 07 IN98 PIC 1 INDIC 98. <-ALL-FMTS
89 +000030 07 IN97 PIC 1 INDIC 97. <-ALL-FMTS
+000031* OUTPUT FORMAT:DTLSCR FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000032* CUSTOMER INQUIRY SCR. #2 <-ALL-FMTS
90 +000033 05 DTLSCR-0 REDEFINES DSPFIL-RECORD. <-ALL-FMTS

```

Figure 10-23 (Part 2 of 11). Source Program Example — CSFMUL (System-Supplied Formats)

```

91 +000034      06 CUSTN          PIC X(6).          <-ALL-FMTS
92 +000035      06 DEPT          PIC S9(3).          <-ALL-FMTS
93 +000036      06 DLSTR         PIC S9(6).          <-ALL-FMTS
94 +000037      06 DSLSM         PIC S9(9).          <-ALL-FMTS
95 +000038      06 DSPM1         PIC S9(9).          <-ALL-FMTS
96 +000039      06 DSPM2         PIC S9(9).          <-ALL-FMTS
97 +000040      06 DSPM3         PIC S9(9).          <-ALL-FMTS
98 +000041      06 DSTYD         PIC S9(11).         <-ALL-FMTS
99 +000042      06 CNAME         PIC X(5).          <-ALL-FMTS
+000043* INPUT FORMAT:ITMMNU FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000044* ITEM INQUIRY SCREEN ONE <-ALL-FMTS
100 +000045      05 ITMMNU-I REDEFINES DSPFIL-RECORD. <-ALL-FMTS
101 +000046      06 ITMMNU-I-INDIC. <-ALL-FMTS
102 +000047      07 IN99          PIC 1 INDIC 99.     <-ALL-FMTS
103 +000048      07 IN98          PIC 1 INDIC 98.     <-ALL-FMTS
104 +000049      07 IN97          PIC 1 INDIC 97.     <-ALL-FMTS
105 +000050      06 ITEMNO        PIC S9(6).          <-ALL-FMTS
+000051* OUTPUT FORMAT:ITMMNU FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000052* ITEM INQUIRY SCREEN ONE <-ALL-FMTS
+000053*      05 ITMMNU-0 REDEFINES DSPFIL-RECORD. <-ALL-FMTS
+000054* INPUT FORMAT:ITMSC2 FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000055* ITEM INQUIRY SCREEN TWO <-ALL-FMTS
106 +000056      05 ITMSC2-I REDEFINES DSPFIL-RECORD. <-ALL-FMTS
107 +000057      06 ITMSC2-I-INDIC. <-ALL-FMTS
108 +000058      07 IN99          PIC 1 INDIC 99.     <-ALL-FMTS
109 +000059      07 IN98          PIC 1 INDIC 98.     <-ALL-FMTS
110 +000060      07 IN97          PIC 1 INDIC 97.     <-ALL-FMTS
+000061* OUTPUT FORMAT:ITMSC2 FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000062* ITEM INQUIRY SCREEN TWO <-ALL-FMTS
111 +000063      05 ITMSC2-0 REDEFINES DSPFIL-RECORD. <-ALL-FMTS
112 +000064      06 DSC          PIC X(30).          <-ALL-FMTS
113 +000065      06 QAVAIL       PIC S9(7).          <-ALL-FMTS
114 +000066      06 QTYH         PIC S9(7).          <-ALL-FMTS
115 +000067      06 QTYO         PIC S9(7).          <-ALL-FMTS
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 5
STMT SEQNBR -A 1 B.+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
116 +000068      06 QTYB         PIC S9(7).          <-ALL-FMTS
117 +000069      06 UNT          PIC X(2).          <-ALL-FMTS
118 +000070      06 PR1          PIC S9(5)V9(2).       <-ALL-FMTS
119 +000071      06 PR5          PIC S9(7).          <-ALL-FMTS
120 +000072      06 UFR          PIC S9(3)V9(2).       <-ALL-FMTS
+000073* INPUT FORMAT:ITMSC3 FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000074* ITEM INQUIRY SCREEN 3 <-ALL-FMTS
121 +000075      05 ITMSC3-I REDEFINES DSPFIL-RECORD. <-ALL-FMTS
122 +000076      06 ITMSC3-I-INDIC. <-ALL-FMTS
123 +000077      07 IN99          PIC 1 INDIC 99.     <-ALL-FMTS
124 +000078      07 IN98          PIC 1 INDIC 98.     <-ALL-FMTS
125 +000079      07 IN97          PIC 1 INDIC 97.     <-ALL-FMTS
+000080* OUTPUT FORMAT:ITMSC3 FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000081* ITEM INQUIRY SCREEN 3 <-ALL-FMTS
126 +000082      05 ITMSC3-0 REDEFINES DSPFIL-RECORD. <-ALL-FMTS
127 +000083      06 SLSM         PIC S9(7)V9(2).       <-ALL-FMTS
128 +000084      06 SLSY         PIC S9(9)V9(2).       <-ALL-FMTS
129 +000085      06 CSTM         PIC S9(7)V9(2).       <-ALL-FMTS
130 +000086      06 CSTY         PIC S9(9)V9(2).       <-ALL-FMTS
131 +000087      06 PROFIT       PIC S9(3)V9(2).       <-ALL-FMTS
132 +000088      06 LOSTS        PIC S9(7)V9(2).       <-ALL-FMTS
+000089* INPUT FORMAT:TIMOUT FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000090* TIME OUT SCREEN <-ALL-FMTS
133 +000091      05 TIMOUT-I REDEFINES DSPFIL-RECORD. <-ALL-FMTS
134 +000092      06 TIMOUT-I-INDIC. <-ALL-FMTS
135 +000093      07 IN99          PIC 1 INDIC 99.     <-ALL-FMTS
136 +000094      07 IN98          PIC 1 INDIC 98.     <-ALL-FMTS
137 +000095      07 IN97          PIC 1 INDIC 97.     <-ALL-FMTS
138 +000096      06 TIMRSP       PIC X(1).          <-ALL-FMTS
+000097* OUTPUT FORMAT:TIMGUT FROM FILE DSPFIL OF LIBRARY ICFLIB <-ALL-FMTS
+000098* TIME OUT SCREEN <-ALL-FMTS
+000099*      05 TIMOUT-0 REDEFINES DSPFIL-RECORD. <-ALL-FMTS
139 009900 FD QPRINT 10/01/87
140 010000 LABEL RECORDS ARE OMITTED. 10/01/87
141 010100 01 PRINTREC. 01/14/88
142 010200 05 RC PIC 9999. 01/15/88
143 010300 05 ERRMSG PIC X(128). 01/14/88
144 010400 WORKING-STORAGE SECTION. 10/01/87
145 010500 77 STATUS-IND PIC X(2). 10/01/87
146 010600 77 STATUS-DSP PIC X(2). 10/01/87
147 010700 77 MAJ-MIN-SAV PIC X(4). 10/01/87
148 010800 77 EOF-PFILE-SW PIC X VALUE "0". 10/01/87
149 010900 77 ERR-SW PIC X VALUE "0". 10/01/87
150 011000 77 INDON PIC 1 VALUE B"1". 10/01/87

```

Figure 10-23 (Part 3 of 11). Source Program Example — CSFMUL (System-Supplied Formats)

```

151 011100 77 INDOFF PIC 1 VALUE B"0". 10/01/87
152 011200 77 OPEN-COUNT PIC 9(1) VALUE 0. 10/01/87
153 011300 77 LEN PIC 9(10)V9(5) COMP. 10/01/87
154 011400 77 PROFM PIC 9(7)V9(2) COMP-4. 10/01/87
155 011500 77 CMD2 PIC X(31) 10/01/87
156 011600 VALUE "CPYF HEXDUMP *LIST PRFTMT(*HEX)". 10/01/87
157 011700 01 TR-CTL-AREA. 10/01/87
158 011800 05 FILLER PIC X(2). 10/01/87
159 011900 05 PGM-DEV-NME PIC X(10). 10/01/87
160 012000 05 RCD-FMT-NME PIC X(10). 10/01/87
161 012100 01 DSPF-INDIC-AREA. 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 6
STMT SEQNBR -A 1 B...+...2...+...3...+...4...+...5...+...6...+...7...IDENTFCN S COPYNAME CHG DATE
162 012200 05 IN23 PIC 1 INDIC 23. 10/01/87
163 012300 88 IN23-ON VALUE B"1". 10/01/87
164 012400 88 IN23-OFF VALUE B"0". 10/01/87
165 012500 05 IN97 PIC 1 INDIC 97. 10/01/87
166 012600 88 IN97-ON VALUE B"1". 10/01/87
167 012700 88 IN97-OFF VALUE B"0". 10/01/87
168 012800 05 IN98 PIC 1 INDIC 98. 10/01/87
169 012900 88 IN98-ON VALUE B"1". 10/01/87
170 013000 88 IN98-OFF VALUE B"0". 10/01/87
171 013100 05 IN99 PIC 1 INDIC 99. 10/01/87
172 013200 88 IN99-ON VALUE B"1". 10/01/87
173 013300 88 IN99-OFF VALUE B"0". 10/01/87
174 013400 01 MAJ-MIN. 10/01/87
175 013500 05 MAJ PIC X(2). 10/01/87
176 013600 05 MIN PIC X(2). 10/01/87
177 013700 01 DISPLAY-FEEDBACK. 10/01/87
178 013800 05 CMD-KEY PIC X(2). 10/01/87
179 013900 05 FILLER PIC X(10). 10/01/87
180 014000 05 RCD-FMT PIC X(10). 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 7
STMT SEQNBR -A 1 B...+...2...+...3...+...4...+...5...+...6...+...7...IDENTFCN S COPYNAME CHG DATE
014100/
181 014200 PROCEDURE DIVISION. 10/01/87
014300 DECLARATIVES. 10/01/87
014400* 2 10/14/87
014500***** 02/22/89
014600* * 02/22/89
014700* AN ERROR ON THE DISPLAY FILE - DSPFIL - MAKES IT INACTIVE * 02/22/89
014800* AN ERROR MESSAGE IS PRINTED, THE FILES ARE CLOSED AND THE * 02/22/89
014900* PROGRAM IS ENDED. * 02/22/89
015000* * 02/22/89
015100***** 02/22/89
015200* 10/08/87
015300 DSP-ERROR SECTION. 10/05/87
015400 USE AFTER STANDARD ERROR PROCEDURE ON DSPFIL. 10/08/87
015500* 10/05/87
015600 DSPFIL-EXCEPTION. 10/08/87
182 015700 MOVE "DISPLAY ERROR. JOB TERMINATED" TO ERRMSG. 02/21/89
183 015800 WRITE PRINTREC. 02/21/89
184 015900 CLOSE CMNFIL DSPFIL QPRINT. 02/21/89
185 016000 STOP RUN. 02/21/89
016100* 10/05/87
016200 CMN-ERROR SECTION. 10/14/87
016300 USE AFTER STANDARD ERROR PROCEDURE ON CMNFIL. 10/08/87
016400 CMNFIL-EXCEPTION. 10/08/87
016500***** 02/22/89
016600* CHECK THE MAJOR/MINOR CODES AND TAKE APPROPRIATE ACTION * 02/22/89
016700* MAJOR CODE 34 IS AN INPUT EXCEPTION * 02/22/89
016800***** 02/22/89
186 016900 IF MAJ-MIN = "3431" 10/01/87
017000* DATA TRUNCATED IN INPUT AREA. SAVE RETURN CODE . 10/01/87
187 017100 MOVE MAJ-MIN TO MAJ-MIN-SAV 10/01/87
188 017200 GO TO EXIT-DECLARATIVES. 10/13/87
017300* 10/08/87
017400* RECOVERABLE SESSION ERROR. CLOSE ICF FILE. 10/03/90
189 017500 IF MAJ = "83" 10/01/87
190 017600 MOVE MAJ-MIN TO RC 01/14/88
191 017700 MOVE "PROGRAM STARTED AGAIN DUE TO SESSION ERROR" 10/01/87
017800 TO ERRMSG 01/14/88
192 017900 WRITE PRINTREC 10/01/87
193 018000 MOVE "1" TO ERR-SW 10/01/87
194 018100 GO TO EXIT-DECLARATIVES. 10/01/87
018200* 10/01/87
018300***** 02/21/89
018400* THIS ROUTINE IS CALLED WHEN THERE IS A PERMANENT SESSION ERROR. * 02/21/89
018500* GET INFORMATION FROM THE MAJOR-MINOR CODE AND PLACE IT INTO * 02/21/89
018600* A DATABASE FILE. THEN PRINT THE FILE IN HEX USING COPYFILE. * 02/21/89

```

Figure 10-23 (Part 4 of 11). Source Program Example — CSFMUL (System-Supplied Formats)

```

018700*****
018800*
018900 GETFBA.
195 019000 MOVE MAJ-MIN TO RC.
196 019100 MOVE "PROGRAM TERMINATED DUE TO ERROR IN CMNFIL FILE"
019200 TO ERRMSG.
197 019300 WRITE PRINTREC.
198 019400 CLOSE CMNFIL DSPFIL
019500 QPRINT.
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 8
STMT SEQNBR -A 1 B...+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
199 019600 STOP RUN.
019700*
019800 EXIT-DECLARATIVES.
019900 EXIT.
020000*
200 020100 END DECLARATIVES.
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 9
STMT SEQNBR -A 1 B...+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
020200/
020300 START-PROGRAM SECTION.
020400*
020500 START-PROGRAM-PARAGRAPH.
020600* 3
201 020700 OPEN I-O CMNFIL DSPFIL
020800 OUTPUT QPRINT.
020900*
021000*****
021100* THE FOLLOWING TEST IS TO ATTEMPT RECOVERY IF AN ERROR OCCURS *
021200* WHEN OPENING THE ICF FILE. *
021300*****
021400* 4
202 021500 IF ERR-SW = "1"
203 021600 THEN IF OPEN-COUNT IS = 9
204 021700 THEN PERFORM DETACH-ROUTINE THRU DETACH-EXIT
205 021800 GO TO END-JOB
021900 ELSE
206 022000 ADD 1 TO OPEN-COUNT
207 022100 PERFORM ERROR-RECOVERY
208 022200 GO TO START-PROGRAM-PARAGRAPH
022300 ELSE
209 022400 MOVE 0 TO OPEN-COUNT.
022500*
022600*****
022700*
022800* THE DISPLAY DEVICE IS IMPLICITLY ACQUIRED WHEN THE
022900* FILE IS OPENED.
023000*
023100* ALL OF THE ICF PROGRAM DEVICES ARE EXPLICITLY ACQUIRED.
023200*
023300* EACH OF THE FOUR TARGET PROGRAMS ARE EVOKED TO ESTABLISH
023400* TRANSACTIONS WITH THE REMOTE SYSTEMS.
023500*
023600* THE MAIN INQUIRY MENU (CIMENU) IS WRITTEN TO THE USER'S
023700* DISPLAY.
023800*
023900* EVOKE PROGRAM "CTFMULCL" ON REMOTE SYSTEM IN LIBRARY ICFLIB.
024000*
024100*****
024200* 5
210 024300 ACQUIRE "ICF00 " FOR CMNFIL.
211 024400 ACQUIRE "ICF01 " FOR CMNFIL.
212 024500 ACQUIRE "ICF02 " FOR CMNFIL.
213 024600 ACQUIRE "ICF03 " FOR CMNFIL.
214 024700 PERFORM EVOKE-ROUTINE THRU EVOKE-EXIT.
024800*
215 024900 WRITE DSPREC FORMAT IS "CIMENU"
025000 INDICATORS ARE DSPF-INDIC-AREA.
025100*****
025200*
025300* DETERMINE USER'S REQUEST
025400*
025500* THIS PORTION OF THE PROGRAM ISSUES A READ TO THE DISPLAY
025600* DEVICE TO RECEIVE THE USER'S REQUEST. THE TYPE
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 10
STMT SEQNBR -A 1 B...+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
025700* OF REQUEST MADE IS BASED ON THE DISPLAY FORMAT CURRENTLY
025800* ON THE SCREEN. THE RECORD FORMAT NAME IS EXTRACTED FROM
025900* THE I/O FEEDBACK AREA OF THE DISPLAY FILE AND USED TO
026000* DETERMINE WHICH ACTION SHOULD BE TAKEN NEXT.

```

Figure 10-23 (Part 5 of 11). Source Program Example — CSFMUL (System-Supplied Formats)

```

026100*
026200*****
026300* 6
026400 READRQ.
216 026500 READ DSPFIL INDICATORS ARE DSPF-INDIC-AREA.
217 026600 IF RCD-FMT = "CIMENU"
218 026700 PERFORM MENU-ROUTINE THRU MENU-EXIT
219 026800 GO TO READRQ.
220 026900 IF RCD-FMT = "ITMMNU"
221 027000 PERFORM ITMIN-ROUTINE THRU ITMIN-EXIT
222 027100 GO TO READRQ.
223 027200 IF RCD-FMT = "ITMSC2"
224 027300 PERFORM ITMRTN-ROUTINE THRU ITMRTN-EXIT
225 027400 GO TO READRQ.
226 027500 IF RCD-FMT = "ITMSC3"
227 027600 PERFORM ITMRTN-ROUTINE THRU ITMRTN-EXIT
228 027700 GO TO READRQ.
229 027800 IF RCD-FMT = "DTLMNU"
230 027900 PERFORM DTLIN-ROUTINE THRU DTLIN-EXIT
231 028000 GO TO READRQ.
232 028100 IF RCD-FMT = "DTLSCR"
233 028200 PERFORM DTLRTN-ROUTINE THRU DTLRTN-EXIT
234 028300 GO TO READRQ.
235 028400 WRITE DSPREC FORMAT IS "CIMENU".
236 028500 GO TO READRQ.
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 11
STMT SEQNBR -A 1 B..+....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
028600/
028700*****
028800*
028900* MAIN MENU
029000*
029100* THE MAIN MENU IS READ TO DETERMINE THE REQUEST ENTERED
029200* BY THE USER. IF CMD 1 IS PRESSED, THE PROGRAM IS ENDED.
029300* IF OPTION = 1, AN ITEM INQUIRY MENU IS WRITTEN TO
029400* THE SCREEN. IF OPTION = 2, A CUSTOMER INQUIRY MENU IS
029500* WRITTEN TO THE SCREEN.
029600*
029700*****
029800* 7
029900 MENU-ROUTINE.
237 030000 IF CMD-KEY = "01"
238 030100 PERFORM DETACH-ROUTINE THRU DETACH-EXIT
239 030200 GO TO END-JOB.
240 030300 IF OPTION = "1"
241 030400 WRITE DSPREC FORMAT IS "ITMMNU"
030500 ELSE
242 030600 WRITE DSPREC FORMAT IS "DTLMNU".
030700 MENU-EXIT.
030800 EXIT.
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 12
STMT SEQNBR -A 1 B..+....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
030900/
031000*****
031100*
031200* ITEM INQUIRY
031300*
031400* THE ITEM NUMBER REQUESTED BY THE USER ON THE ITEM INQUIRY
031500* SCREEN IS CHECKED. THIS IS DETERMINED BY THE
031600* DISPLAY RECORD FORMAT BEING PROCESSED - IN THIS CASE ITMMNU.
031700*
031800* IF CMD KEY 1 IS PRESSED, THE PROGRAM IS ENDED. IF CMD KEY 2
031900* IS PRESSED, THE ITEM INQUIRY REQUEST IS CANCELED, AND THE
032000* MAIN MENU (CIMENU) IS WRITTEN TO THE SCREEN.
032100*
032200* IF AN ITEM NUMBER IS ENTERED, A ITEM INQUIRY REQUEST IS
032300* SENT TO THE APPROPRIATE REMOTE SYSTEM. THE REMOTE SYSTEM
032400* IS SELECTED BASED ON THE ITEM NUMBER REQUESTED.
032500*
032600* A CHECK IS MADE FOR THREE CONDITIONS FOLLOWING THE READ.
032700* 1) THE REMOTE SYSTEM TIMED OUT, 2) NO DATA RECEIVED, AND
032800* 3) DATA RETURNED IN AN UNEXPECTED RECORD FORMAT.
032900*
033000* IF THE REMOTE SYSTEM TIMES OUT (MAJ-MIN = 0310) A MESSAGE
033100* IS WRITTEN TO THE SCREEN, ASKING TO TRY AGAIN OR END THE
033200* PROGRAM.
033300*
033400* IF NO DATA IS RECEIVED AFTER THE READ OPERATION TO THE
033500* PROGRAM DEVICE (MAJ-MIN = 03__ ) THE REQUEST IS SENT AGAIN
033600* TO THE REMOTE SYSTEM AND THE READ OPERATION IS ISSUED TO

```

Figure 10-23 (Part 6 of 11). Source Program Example — CSFMUL (System-Supplied Formats)

```

033700*   THE PROGRAM DEVICE.                                     *           10/15/87
033800*                                     *           10/14/87
033900*   IF THE RECORD RETURNS WITH THE WRONG RECORD FORMAT, THE *           10/14/87
034000*   PROGRAM WILL GO TO EXIT-FORMAT-ERR ROUTINE.           *           10/14/87
034100*                                     *           10/14/87
034200*****
034300* 8                                     *           10/13/87
243 034400 ITMIN-ROUTINE.                                       *           10/14/87
244 034500   IF CMD-KEY = "01"                                     *           10/01/87
245 034600   PERFORM DETACH-ROUTINE THRU DETACH-EXIT           *           10/08/87
246 034700   GO TO END-JOB.                                     *           10/08/87
247 034800   IF CMD-KEY = "02"                                     *           10/08/87
248 034900   WRITE DSPREC FORMAT IS "CIMENU"                   *           10/01/87
249 035000   GO TO ITMIN-EXIT.                                   *           10/01/87
250 035100   MOVE CORR ITMMNU-I TO ITMREQ.                     *           10/01/87
*                                     *
*   ** CORRESPONDING items for statement 250:                 *
*   ** ITEMNO                                           *
*   ** End of CORRESPONDING items for statement 250
251 035200   IF ITEMNO OF ITMMNU-I LESS THAN 399999 GO TO XICF01. 10/01/87
253 035300   IF ITEMNO OF ITMMNU-I LESS THAN 699999 GO TO XICF02. 10/01/87
255 035400   IF ITEMNO OF ITMMNU-I LESS THAN 899999 GO TO XICF03. 10/01/87
035500 XICF01.                                                  10/01/87
257 035600   MOVE "ICF01 " TO PGM-DEV-NME.                    10/01/87
258 035700   GO TO XITMIN.                                       10/01/87
035800 XICF02.                                                  10/01/87
259 035900   MOVE "ICF02 " TO PGM-DEV-NME.                    10/01/87
260 036000   GO TO XITMIN.                                       10/01/87
5738CB1 V2R1M0 910524      AS/400 COBOL Source      ICFLIB/CSFMUL      RCHAS400 10/03/90 14:37:21   Page 13
STMT SEQNBR -A 1 B.+. . . . 2. . . . 3. . . . 4. . . . 5. . . . 6. . . . 7. . IDENTFCN  S COPYNAME  CHG DATE
036100 XICF03.                                                  10/01/87
261 036200   MOVE "ICF03 " TO PGM-DEV-NME.                    10/01/87
036300 XITMIN.                                                  10/01/87
262 036400   MOVE 150 TO LNTH OF ITMREQ.                       02/22/89
263 036500   WRITE CMNREC FORMAT IS "$$SEND"                   10/01/87
036600   TERMINAL IS PGM-DEV-NME.                              10/01/87
036700 RETRY-ITEM.                                             10/01/87
264 036800   READ CMNFIL.                                       10/08/87
265 036900   IF MAJ-MIN = "0310"                                10/01/87
266 037000   WRITE DSPREC FORMAT IS "TIMOUT"                   10/01/87
267 037100   READ DSPFIL INDICATORS ARE DSPF-INDIC-AREA       10/08/87
268 037200   IF TIMRSP = "1" GO TO RETRY-ITEM-END-IF          01/21/88
270 037300   IF TIMRSP = "2" GO TO END-JOB-END-IF.           01/21/88
272 037400   IF MAJ = "03"                                     10/01/87
273 037500   GO TO XITMIN.                                       10/01/87
274 037600   IF RCD-FMT-NME IS NOT EQUAL "ITMRSP" GO TO EXIT-FORMAT-ERR. 10/02/87
276 037700   PERFORM ITMOUT-ROUTINE THRU ITMOUT-EXIT.         10/01/87
037800 ITMIN-EXIT.                                             10/01/87
037900 EXIT.                                                   10/01/87
5738CB1 V2R1M0 910524      AS/400 COBOL Source      ICFLIB/CSFMUL      RCHAS400 10/03/90 14:37:21   Page 14
STMT SEQNBR -A 1 B.+. . . . 2. . . . 3. . . . 4. . . . 5. . . . 6. . . . 7. . IDENTFCN  S COPYNAME  CHG DATE
038000/
038100*****
038200*                                     *           10/01/87
038300*   PROCESS ITEM INFORMATION                               *           10/01/87
038400*                                     *           10/01/87
038500*   THIS SECTION PROCESSES THE ITEM RECORD RECEIVED FROM THE *           10/01/87
038600*   TARGET PROGRAM AND THE INFORMATION ABOUT THE ITEM IS *           02/21/89
038700*   DISPLAYED. IF ITEMNO IS 0 OR LESS, IT IS AN INVALID REQUEST *           10/15/87
038800*   AND A FRESH ITEM MENU IS WRITTEN TO THE SCREEN. IF THE *           10/15/87
038900*   REQUEST IS VALID, VALUES ARE CALCULATED BASED ON THE *           10/15/87
039000*   INFORMATION RECEIVED.                               *           10/15/87
039100*                                     *           10/01/87
039200*****
039300* 9                                     *           10/14/87
277 039400 ITMOUT-ROUTINE.                                       *           10/01/87
278 039500   IF ITEMNO OF ITMRSP NOT GREATER THAN 0           *           10/01/87
279 039600   WRITE DSPREC FORMAT IS "ITMMNU"                   *           10/01/87
280 039700   GO TO ITMOUT-EXIT.                                   *           10/01/87
281 039800   MOVE QTYLST TO QAVAIL OF ITMSC2-0.                 *           10/01/87
282 039900   MOVE QTY00 TO QTY0 OF ITMSC2-0.                   *           10/01/87
283 040000   MOVE QTY0H TO QTYH OF ITMSC2-0.                   *           10/01/87
284 040100   MOVE QTYB0 TO QTYB OF ITMSC2-0.                   *           10/01/87
285 040200   MOVE UNITQ TO UNT OF ITMSC2-0.                     *           10/01/87
286 040300   MOVE PR01 TO PR1 OF ITMSC2-0.                     *           10/01/87
287 040400   MOVE PR05 TO PR5 OF ITMSC2-0.                     *           10/01/87
288 040500   MOVE UFRT TO UFR OF ITMSC2-0.                     *           10/01/87
289 040600   MOVE DESC TO DSC OF ITMSC2-0.                     *           10/07/87
290 040700   WRITE DSPREC FORMAT IS "ITMSC2".                   *           10/07/87
040800 ITMOUT-EXIT.                                             10/01/87
040900 EXIT.                                                   10/01/87

```

Figure 10-23 (Part 7 of 11). Source Program Example — CSFMUL (System-Supplied Formats)


```

041000* 10/14/87
041100***** 02/21/89
041200* * 02/21/89
041300* ADDITIONAL ITEM INFORMATION * 02/21/89
041400* * 02/21/89
041500* ADDITIONAL ITEM INFORMATION IS PROCESSED AND THE RESULT * 02/21/89
041600* DISPLAYED ON THE SCREEN WHEN A RESPONSE IS READ FROM THE * 02/21/89
041700* DISPLAY STATION WITH AN ITEM SCREEN RECORD FORMAT. * 02/21/89
041800* * 02/21/89
041900* IF CMD KEY 1 IS PRESSED, THE PROGRAM IS ENDED. IF CMD KEY 2 * 02/21/89
042000* IS PRESSED, THE ITEM INQUIRY IS ENDED, AND THE MAIN MENU * 02/21/89
042100* (CIMENU) IS WRITTEN TO THE SCREEN. IF CMD KEY 3 IS PRESSED, * 02/21/89
042200* THE ITEM INQUIRY MENU IS WRITTEN TO THE SCREEN. BY PRESSING * 02/21/89
042300* ENTER WHEN SCREEN 2 IS DISPLAYED, MORE INFORMATION (PROFIT- * 02/21/89
042400* LOSS) WILL BE DISPLAYED TO THE SCREEN. WHEN SCREEN 3 IS * 02/21/89
042500* DISPLAYED, AN ENTER WILL CAUSE THE ITEM INQUIRY MENU * 02/21/89
042600* TO BE WRITTEN TO THE SCREEN. * 02/21/89
042700* * 02/21/89
042800***** 02/21/89
042900* 10 10/14/87
291 043000 ITMRTN-ROUTINE. 10/01/87
292 043100 IF CMD-KEY = "01" 10/08/87
293 043200 PERFORM DETACH-ROUTINE THRU DETACH-EXIT 10/08/87
294 043300 GO TO END-JOB. 10/08/87
295 043400 IF CMD-KEY = "02" 10/08/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 15
STMT SEQNBR -A 1 B..+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
296 043500 WRITE DSPREC FORMAT IS "CIMENU" 10/01/87
297 043600 GO TO ITMRTN-EXIT. 10/01/87
298 043700 IF CMD-KEY = "03" 10/08/87
299 043800 WRITE DSPREC FORMAT IS "ITMMNU" 10/08/87
300 043900 GO TO ITMRTN-EXIT. 10/08/87
301 044000 IF RCD-FMT = "ITMSC2" 10/08/87
302 044100 PERFORM PROFIT-LOSS THRU PROFIT-LOSS-EXIT 10/01/87
303 044200 WRITE DSPREC FORMAT IS "ITMSC3" 10/01/87
304 044300 GO TO ITMRTN-EXIT. 10/01/87
305 044400 WRITE DSPREC FORMAT IS "ITMMNU". 10/01/87
044500 ITMRTN-EXIT. 10/01/87
044600 EXIT. 10/01/87
044700* 10/01/87
044800***** 02/21/89
044900* * 02/21/89
045000* THIS SECTION OF THE PROGRAM IS CALLED TO PROCESS * 02/21/89
045100* THE INFORMATION FOR THE ITEM NUMBER REQUESTED BEFORE * 02/21/89
045200* IT IS SENT BACK TO THE REQUESTING REMOTE SYSTEM. * 02/21/89
045300* * 02/21/89
045400***** 02/21/89
045500* 11 10/14/87
045600* 10/01/87
306 045700 PROFIT-LOSS. 10/01/87
307 045800 SUBTRACT SLSTM FROM CSTTM GIVING PROFM. 10/01/87
308 045900 MULTIPLY PROFM BY 100 GIVING PROFM. 10/01/87
309 046000 IF SLSTM GREATER THAN 0 10/01/87
310 046100 DIVIDE PROFM BY SLSTM GIVING PROFM. 10/01/87
311 046200 MULTIPLY QTYLST BY PR01 GIVING LOSTS. 10/01/87
312 046300 MOVE SLSTM TO SLSM. 10/01/87
313 046400 MOVE SLSTY TO SLSY. 10/01/87
314 046500 MOVE CSTTM TO CSTM. 10/01/87
315 046600 MOVE PROFM TO PROFIT. 10/01/87
316 046700 MOVE CSTTY TO CSTY. 10/01/87
046800 PROFIT-LOSS-EXIT. 10/01/87
046900 EXIT. 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 16
STMT SEQNBR -A 1 B..+...2...+...3...+...4...+...5...+...6...+...7..IDENTFCN S COPYNAME CHG DATE
047000/ 10/01/87
047100***** 10/13/87
047200* * 10/13/87
047300* CUSTOMER INQUIRY * 10/13/87
047400* * 10/13/87
047500* THE REQUEST FROM THE CUSTOMER INQUIRY MENU IS PROCESSED. * 10/13/87
047600* IF CMD KEY 1 IS PRESSED, THE PROGRAM IS ENDED. IF CMD KEY 2 * 10/13/87
047700* IS PRESSED, THE MAIN MENU (CIMENU) IS WRITTEN TO THE SCREEN. * 10/13/87
047800* * 10/13/87
047900* IF A CUSTOMER NUMBER IS ENTERED, THE CUSTOMER INQUIRY * 10/13/87
048000* REQUEST IS SENT TO THE REMOTE SYSTEM. THEN DTOUT-ROUTINE * 10/13/87
048100* THRU DTOUT-EXIT ARE PERFORMED. * 10/14/87
048200* * 10/13/87
048300***** 10/13/87
048400* 12 10/14/87
317 048500 DTLIN-ROUTINE. 10/01/87

```

Figure 10-23 (Part 8 of 11). Source Program Example — CSFMUL (System-Supplied Formats)

```

318 048600 IF CMD-KEY = "01" 10/08/87
319 048700 PERFORM DETACH-ROUTINE THRU DETACH-EXIT 10/08/87
320 048800 GO TO END-JOB. 10/08/87
321 048900 IF CMD-KEY = "02" 10/08/87
322 049000 WRITE DSPREC FORMAT IS "CIMENU" 10/08/87
323 049100 GO TO DTLIN-EXIT. 10/01/87
049200 EVDTL. 10/01/87
324 049300 MOVE 150 TO LENGTH OF DTLREQ. 02/22/89
325 049400 MOVE "ICF00 " TO PGM-DEV-NME. 10/01/87
326 049500 MOVE CORR DTLMNU-I TO DTLREQ. 10/01/87
* ** CORRESPONDING items for statement 326:
* ** CUSTNO
* ** End of CORRESPONDING items for statement 326
327 049600 WRITE CMNREC FORMAT IS "$$SEND" 10/01/87
049700 TERMINAL IS PGM-DEV-NME. 10/01/87
328 049800 PERFORM DTOUT-ROUTINE THRU DTOUT-EXIT. 10/01/87
049900 DTLIN-EXIT. 10/01/87
050000 EXIT. 10/01/87
050100* 10/14/87
050200***** 10/13/87
050300* 10/13/87
050400* PROCESS CUSTOMER INFORMATION * 10/13/87
050500* * 10/13/87
050600* THE CUSTOMER DATA RECEIVED FROM THE TARGET PROGRAM IS * 10/13/87
050700* PROCESSED. IF CUSTOMER NUMBER IS ZERO OR LESS, IT IS AN * 10/13/87
050800* INVALID REQUEST AND THE MAIN MENU IS WRITTEN TO THE SCREEN. * 10/13/87
050900* * 10/13/87
051000***** 10/13/87
051100* 13 10/14/87
329 051200 DTOUT-ROUTINE. 10/01/87
330 051300 IF CUSTNO OF DTLRSP NOT GREATER THAN 0 10/01/87
331 051400 WRITE DSPREC FORMAT IS "CIMENU" 10/01/87
332 051500 GO TO DTOUT-EXIT. 10/01/87
333 051600 PERFORM CUSTOMER-DETAIL THRU CUSTOMER-DETAIL-EXIT. 10/07/87
051700 DTOUT-EXIT. 10/01/87
051800 EXIT. 10/01/87
051900* 10/01/87
052000***** 02/21/89
052100* * 02/21/89
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 17
STMT SEQNBR -A 1 B..+...2....+...3....+...4....+...5....+...6....+...7..IDENTFCN S COPYNAME CHG DATE
052200* THIS SECTION OF THE PROGRAM HANDLES THE REQUEST FOLLOWING * 02/21/89
052300* THE DISPLAY OF THE CUSTOMER INFORMATION. CMD KEY 1 WILL * 02/21/89
052400* EXIT THE JOB, CMD KEY 2 WILL DISPLAY THE MAIN MENU, AND * 02/21/89
052500* AN "ENTER" WILL BRING UP THE CUSTOMER INQUIRY MENU. * 02/21/89
052600* * 02/21/89
052700***** 02/21/89
052800* 14 10/14/87
334 052900 DTLRTN-ROUTINE. 10/08/87
335 053000 IF CMD-KEY = "01" 10/08/87
336 053100 PERFORM DETACH-ROUTINE THRU DETACH-EXIT 10/08/87
337 053200 GO TO END-JOB. 10/08/87
338 053300 IF CMD-KEY = "02" 10/08/87
339 053400 WRITE DSPREC FORMAT IS "CIMENU" 10/08/87
340 053500 GO TO DTLRTN-EXIT. 10/08/87
341 053600 WRITE DSPREC FORMAT IS "DTLMNU". 10/08/87
053700 DTLRTN-EXIT. 10/08/87
053800 EXIT. 10/08/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 18
STMT SEQNBR -A 1 B..+...2....+...3....+...4....+...5....+...6....+...7..IDENTFCN S COPYNAME CHG DATE
053900/ 10/14/87
054000***** 02/21/89
054100* * 02/21/89
054200* THE READ OPERATION TO THE PROGRAM DEVICE IS ISSUED. * 02/21/89
054300* A CHECK IS MADE FOR THREE CONDITIONS FOLLOWING THE READ. * 02/21/89
054400* 1) THE REMOTE SYSTEM TIMED OUT. 2) NO DATA RECEIVED. AND * 02/21/89
054500* 3) DATA RETURNED IN AN UNEXPECTED RECORD FORMAT. * 02/21/89
054600* * 02/21/89
054700* IF THE REMOTE SYSTEM TIMES OUT (MAJ-MIN = 0310) A MESSAGE * 02/21/89
054800* IS WRITTEN TO THE SCREEN, ASKING TO TRY AGAIN OR END THE * 02/21/89
054900* PROGRAM. * 02/21/89
055000* * 02/21/89
055100* IF NO DATA IS RECEIVED AFTER THE READ OPERATION TO THE * 02/21/89
055200* PROGRAM DEVICE (MAJ-MIN = 03__ ) THE REQUEST IS SENT AGAIN * 02/21/89
055300* TO THE REMOTE SYSTEM AND THE READ OPERATION IS ISSUED TO * 02/21/89
055400* THE ICF PROGRAM DEVICE. * 10/03/90
055500* * 02/21/89
055600* IF THE RECORD RETURNS WITH THE WRONG RECORD FORMAT, THE * 02/21/89
055700* PROGRAM WILL GO TO EXIT-FORMAT-ERR ROUTINE. * 02/21/89
055800***** 02/21/89

```

Figure 10-23 (Part 9 of 11). Source Program Example — CSFMUL (System-Supplied Formats)

```

055900* 15 10/14/87
056000* 10/01/87
342 056100 CUSTOMER-DETAIL. 10/01/87
343 056200 READ CMNFIL. 10/08/87
344 056300 IF MAJ-MIN = "0310" 10/01/87
345 056400 WRITE DSPREC FORMAT IS "TIMEOUT" 10/01/87
346 056500 READ DSPFIL INDICATORS ARE DSPF-INDIC-AREA 10/08/87
347 056600 IF TIMRSP = "1" GO TO CUSTOMER-DETAIL END-IF 01/21/88
349 056700 IF TIMRSP = "2" GO TO END-JOB END-IF. 01/21/88
351 056800 IF MAJ = "03" 10/01/87
352 056900 WRITE CMNREC FORMAT IS "$$SEND" 10/01/87
057000 TERMINAL IS PGM-DEV-NME 10/01/87
353 057100 GO TO CUSTOMER-DETAIL. 10/01/87
354 057200 IF RCD-FMT-NME IS NOT EQUAL "DTLRSP" GO TO EXIT-FORMAT-ERR. 10/02/87
356 057300 MOVE CUSTNO OF DTLRSP TO CUSTN OF DTLSRCR-0. 10/07/87
357 057400 MOVE DNAME OF DTLRSP TO CNAME OF DTLSRCR-0. 03/21/89
358 057500 MOVE DLSTOR OF DTLRSP TO DLSTR OF DTLSRCR-0. 10/07/87
359 057600 MOVE DSLSTM OF DTLRSP TO DSLSM OF DTLSRCR-0. 10/07/87
360 057700 MOVE DSPM01 OF DTLRSP TO DSPM1 OF DTLSRCR-0. 10/07/87
361 057800 MOVE DSPM02 OF DTLRSP TO DSPM2 OF DTLSRCR-0. 10/07/87
362 057900 MOVE DSPM03 OF DTLRSP TO DSPM3 OF DTLSRCR-0. 10/07/87
363 058000 MOVE DSTTYD OF DTLRSP TO DSTYD OF DTLSRCR-0. 10/07/87
364 058100 MOVE IDEPT OF DTLRSP TO DEPT OF DTLSRCR-0. 10/07/87
365 058200 WRITE DSPREC FORMAT IS "DTLSRCR". 10/07/87
058300 CUSTOMER-DETAIL-EXIT. 10/01/87
058400 EXIT. 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 19
STMT SEQNBR -A 1 B...+...2...+...3...+...4...+...5...+...6...+...7...IDENTFCN S COPYNAME CHG DATE
058500/ 10/01/87
058600***** 02/21/89
058700* * 02/21/89
058800* THIS SUBROUTINE IS CALLED TO EVOKE THE TARGET PROGRAM. * 02/21/89
058900* THE SAME TARGET PROGRAM (ICFLIB/CTFMULCL) IS EVOKED AT * 02/21/89
059000* FOUR DIFFERENT REMOTE SYSTEMS. THE PROGRAM DEVICE * 02/21/89
059100* IDENTIFIES WHICH SESSION SHOULD BE EVOKED. THE PROGRAM * 02/21/89
059200* DEVICE WAS SPECIFIED IN CMID PRIOR TO CALLING THIS ROUTINE. * 02/21/89
059300* * 02/21/89
059400***** 02/21/89
059500* 16 10/14/87
059600* 10/01/87
366 059700 EVOKE-ROUTINE. 10/01/87
367 059800 MOVE 0 TO LNGTH OF EVKREQ. 02/22/89
368 059900 MOVE "CTFMULCL" TO PGMJD OF EVKREQ. 10/01/87
369 060000 MOVE "ICFLIB" TO LIB OF EVKREQ. 10/08/87
370 060100 MOVE "ICF00 " TO PGM-DEV-NME 10/01/87
371 060200 WRITE CMNREC FORMAT IS "$$EVOKNI" 10/01/87
060300 TERMINAL IS PGM-DEV-NME. 10/01/87
372 060400 MOVE "ICF01 " TO PGM-DEV-NME 10/01/87
373 060500 WRITE CMNREC FORMAT IS "$$EVOKNI" 10/01/87
060600 TERMINAL IS PGM-DEV-NME. 10/01/87
374 060700 MOVE "ICF02 " TO PGM-DEV-NME 10/01/87
375 060800 WRITE CMNREC FORMAT IS "$$EVOKNI" 10/01/87
060900 TERMINAL IS PGM-DEV-NME. 10/01/87
376 061000 MOVE "ICF03 " TO PGM-DEV-NME 10/01/87
377 061100 WRITE CMNREC FORMAT IS "$$EVOKNI" 10/01/87
061200 TERMINAL IS PGM-DEV-NME. 10/01/87
061300 EVOKE-EXIT. 10/01/87
061400 EXIT. 10/01/87
061500* 10/01/87
061600***** 02/21/89
061700* * 02/21/89
061800* THE TRANSACTION AND SESSION ARE ENDED FOR EACH OF THE * 02/21/89
061900* REMOTE SYSTEMS. * 02/21/89
062000* * 02/21/89
062100***** 02/21/89
062200* 17 10/14/87
378 062300 ERROR-RECOVERY. 10/01/87
379 062400 PERFORM DETACH-ROUTINE THRU DETACH-EXIT. 10/01/87
380 062500 CLOSE CMNFIL DSPFIL 10/08/87
062600 QPRINT. 10/01/87
381 062700 MOVE "0" TO ERR-SW. 10/01/87
062800 ERROR-RECOVERY-EXIT. 10/01/87
062900 EXIT. 10/01/87
063000***** 02/21/89
063100* * 02/21/89
063200* EXIT-FORMAT-ERR IS BRANCHED TO AFTER A READ TO CMNFIL. THE * 02/21/89
063300* RCD-FMT-NME RETURNED IN THE I-0-FEEDBACK AREA DOES NOT MATCH * 02/21/89
063400* THE FORMAT EXPECTED BY THE PROGRAM. AN ERROR MESSAGE IS * 02/21/89
063500* PRINTED AND THE PROGRAM ENDS. * 02/21/89
063600* * 02/21/89

```

Figure 10-23 (Part 10 of 11). Source Program Example — CSFMUL (System-Supplied Formats)

```

063700*****
063800* 18
382 063900 EXIT-FORMAT-ERR.
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 20
STMT SEQNBR -A 1 B...2...3...4...5...6...7..IDENTFCN S COPYNAME CHG DATE
383 064000 MOVE MAJ-MIN TO RC.
384 064100 MOVE "RECORD FORMAT IS INCORRECT ON READ "
064200 TO ERRMSG.
385 064300 WRITE PRINTREC.
386 064400 CLOSE CMNFIL DSPFIL QPRINT.
387 064500 STOP RUN.
064600*
064700*****
064800* *
064900* THIS SECTION OF THE PROGRAM IS CALLED TO END *
065000* THE TRANSACTION WITH THE REMOTE SYSTEM. *
065100* *
065200*****
065300* 19
065400 DETACH-ROUTINE.
388 065500 MOVE 0 TO LNTH OF ITMREQ.
389 065600 MOVE "ICF00 " TO PGM-DEV-NME
390 065700 WRITE CMNREC FORMAT IS "$$SENDET"
065800 TERMINAL IS PGM-DEV-NME.
391 065900 MOVE "ICF01 " TO PGM-DEV-NME
392 066000 WRITE CMNREC FORMAT IS "$$SENDET"
066100 TERMINAL IS PGM-DEV-NME.
393 066200 MOVE "ICF02 " TO PGM-DEV-NME
394 066300 WRITE CMNREC FORMAT IS "$$SENDET"
066400 TERMINAL IS PGM-DEV-NME.
395 066500 MOVE "ICF03 " TO PGM-DEV-NME
396 066600 WRITE CMNREC FORMAT IS "$$SENDET"
066700 TERMINAL IS PGM-DEV-NME.
066800 DETACH-EXIT.
066900 EXIT.
067000*
067100*****
067200* *
067300* THIS SECTION OF THE PROGRAM IS CALLED TO RELEASE THE PROGRAM *
067400* DEVICES, END THE SESSION AND END THE PROGRAM. *
067500* *
067600*****
067700* 20
067800*
397 067900 END-JOB.
398 068000 DROP "ICF00 " FROM CMNFIL.
399 068100 DROP "ICF01 " FROM CMNFIL.
400 068200 DROP "ICF02 " FROM CMNFIL.
401 068300 DROP "ICF03 " FROM CMNFIL.
402 068400 CLOSE CMNFIL DSPFIL QPRINT.
403 068500 STOP RUN.
068600*
***** END OF SOURCE *****
5738CB1 V2R1M0 910524 AS/400 COBOL Messages ICFLIB/CSFMUL RCHAS400 10/03/90 14:37:21 Page 21
STMT
* 71 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 009800
Message . . . . : No OUTPUT fields found for format CIMENU.
* 71 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 009800
Message . . . . : No OUTPUT fields found for format DTLMNU.
* 71 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 009800
Message . . . . : No OUTPUT fields found for format ITMMNU.
* 71 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 009800
Message . . . . : No OUTPUT fields found for format TIMOUT.
***** END OF MESSAGES *****
Message Summary
Total Info(0-4) Warning(5-19) Error(20-29) Severe(30-39) Terminal(40-99)
4 0 4 0 0 0
Source records read . . . . . : 686
Copy records read . . . . . : 99
Copy members processed . . . . . : 1
Sequence errors . . . . . : 0
Highest severity message issued . . : 10
LBL0901 00 Program CSFMUL created in library ICFLIB.
***** END OF COMPILATION *****

```

Figure 10-23 (Part 11 of 11). Source Program Example — CSFMUL (System-Supplied Formats)

Target Program Multiple-Session Inquiry

(Example II): The following describes the COBOL target program for multiple-session inquiry program example.

Program Files: The COBOL multiple-session target program uses the following files:

CFILE

A ICF file used to send records to and receive records from the source program. It is done with the file-level

INDARA DDS keyword, indicating a separate indicator area.

PFILE

A database file used to retrieve the record for the item requested from the remote system.

QPRINT

A printer file used to print error messages resulting from communications errors.

DDS Source: The DDS for the ICF file (CFILE) is illustrated in Figure 10-24.

```
5714PW1 R01M00 910524          SEU SOURCE LISTING          10/14/87 17:20:35          PAGE 1
SOURCE FILE . . . . . QICFPUB/ICFLIB
MEMBER . . . . . CFILE
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100  A*****
200  A*
300  A*          ICF FILE
400  A*          USED IN TARGET MULTIPLE SESSION PROGRAM
500  A*
600  A*****
700  A          INDARA
800  A 05        RQSWRT
900  A 10        ALWVRT
1000 A          INDTXT(10 '10 END TRANS.')
1100 A 15        EOS
1200 A 20        FAIL
1300 A          INDTXT(20 '20 F ABORT ST')
1400 A          RCVFAIL(25 'RECEIVED FAIL')
1500 A 30        DETACH
1600 A          INDTXT(30 '30>DETACH TGT')
1700 A          RCVDETACH(44 'RCV DETACH')
1800 A          R SNDPART
1900 A          INVITE
2000 A          RECTYP      1
2100 A          ITEMNO     6
2200 A          EDATA      130
2300 A          FILL1      13
2400 A          R RCVPART
2500 A          RECID2     6
2600 A          PARTDS     80
2700 A          FILL4      64
2800 A          R RCVTRND
2900 A          RCVTRNRND(40 'END OF TRN')
                * * * * * E N D O F S O U R C E * * * * *
```

Figure 10-24. DDS Source for ICF File Used in Target Program Multiple Session Inquiry

The DDS source for the database file (PFILE) is illustrated in Figure 10-25.

```
5714PW1 R01M00 910524          SEU SOURCE LISTING          10/16/87 07:43:14          PAGE 1
SOURCE FILE . . . . . QICFPUB/ICFLIB
MEMBER . . . . . PFILE
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100  A          LIFO
200  A          R DBREC
300  A          RECCUS     1
400  A          DBSEQ      6
500  A          DBDATA    130
600  A          DBFILL    13
700  A          K DBSEQ
                * * * * * E N D O F S O U R C E * * * * *
```

Figure 10-25. DDS Source for Database File Used in Target Program Multiple Session Inquiry

The command needed to create the ICF file is:

```
CRTICFF FILE(ICFLIB/CFILE) SRCFILE(ICFLIB/QICFPUB)
SRCMBR(CFILE) ACQPGMDEV(RQSDEV) TEXT("TARGET ICF
FILE FOR MULTIPLE SESSION PROGRAM")
```

The command needed to define the program device entry is:

```
OVRICFDEVE PGMDEV(RQSDEV) RMTLOCNAME(*REQUESTER)
```

Program Explanation: The following explains the structure of the program examples illustrated in Figure 10-26 on page 10-63 and Figure 10-27 on page 10-68. The ICF file used in the first example is defined by the user, and uses externally described data formats (DDS). The second example uses the same file, but uses program-described data and system-supplied formats. The reference letters in the example below correspond to those in the following program examples.

Although the basic structure of the two examples provided is the same, there are differences because of the way the user-defined formats and the system-supplied formats are used. All output operations to the ICF file in the first example are done using the the WRITE statement with the record format name coded as operand. The output operations to the ICF file in the second example using system-supplied formats are issued with the name of the system-supplied format coded as a literal operand.

Differences between the first and second example are described as notes in each of the following descriptions where necessary.

1 This section defines the ICF file (CFILE) and the database file (PFILE) used in the program.

CFILE is the ICF file used to send records to and receive records from the remote system.

MAJ-MIN is the variable name used to check for the ICF file return codes.

CMNF-INDIC-AREA is the indicator area used with the ICF file to choose options on DDS keywords and operations, and receive response indicators on input operations.

Note: In the program using system-supplied formats, the input records for CFILE are explicitly coded in the program since CFILE is now treated as a program-

described file. The system-supplied file, QICDMF, could have been used instead of CFILE. Using system-supplied files can be done by specifying QICDMF in the file specification, or by using an OVRICFF command to change the file name from CFILE to QICDMF.

2 This section defines the error handling for the program. The routine first checks the major/minor return code to determine if the error is recoverable. If return code 3431 is received, it is saved, and control is passed back to the main calling program. Return code 3431 is not considered a serious error in this example. The program then exits the declaratives.

If any other error has occurred, the program prints a message saying that the program ended abnormally, and then ends.

3 This routine opens all the files.

Because the ICF file was created using the ACQPGMDEV parameter, the target session is automatically acquired when the file is opened.

4 This routine reads data from the program device (CFILE) through a perform statement until a change direction is received. The program then goes to **5** to read the database file. When change direction is received, indicator 40 is set on, as defined by the RCVTRNRND DDS keyword in the DDS source file for ICF file.

Note: In the program using system-supplied formats, a minor return code of '00' is checked to determine if change direction has been received.

5 The program uses the requested number received from the source program to access the record from the database. The information retrieved from the database file (PFILE) is moved to the work area for the ICF file. A write operation is issued to the program device using record format SNDPART. The write operation sends the requested information back to the source program.

If the requested number is not found, zero is propagated into the field.

If an error occurs on the write operation, control passes to **2**.

If no error occurs on the write, control goes back to **4**.

Note: In the program using the system-supplied format, the WRITE statement uses the \$\$\$SEND format to send the data.

- 6** A read operation is issued to the program device.

If a detach indication is received, the program goes to **8** to end the program. When a detach is received, indicator 44 is set on, as defined by the RCVDETACH keyword in the DDS for the ICF file.

Note: In the system-supplied format example, the read operation is sent without using a record format name. Also, a minor return code of '08' is checked for the detach received condition.

- 7** This routine is called to issue the read operation to the the program device until the RCVTRNRND indication is received.

Note: In the system-supplied format example, there is no **7**. Instead, a minor return code of '00' is checked for the turn-around indication in **4**.

- 8** This routine is called to end the program.

The following message is written to the print file:

```
CTDMUL HAS COMPLETED NORMALLY
```

The files are closed. The program device is automatically released as a result of the close operation, and the program ends.

Note: In the program using system-supplied formats, the following message is printed:

```
CTFMUL HAS COMPLETED NORMALLY
```

```
5738CB1 V2R1M0 910524          IBM AS/400 COBOL/400          ICFLIB/CTDMUL          RCHAS400 10/03/90 14:34:16          Page 1
Program . . . . . : CTFMUL
Library . . . . . : ICFLIB
Source file . . . . . : QICFPUB
Library . . . . . : ICFLIB
Source member . . . . . : CTFMUL 10/03/90 14:30:28
Generation severity level . . . . . : 29
Text 'description' . . . . . : CBL Multiple Session Inquiry - Target DDS
Source listing options . . . . . : *SOURCE
Generation options . . . . . : *NONE
Message limit:
  Number of messages . . . . . : *NOMAX
  Message limit severity . . . . . : 29
Print file . . . . . : QSYSVRT
Library . . . . . : *LIBL
FIPS flagging . . . . . : *NOFIPS *NOSEG *NODEB *NOOBSOLETE
SAA flagging . . . . . : *NOFLAG
Flagging severity . . . . . : 0
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LJBCRTAUT
Compiler . . . . . : IBM AS/400 COBOL/400
5738CB1 V2R1M0 910524          AS/400 COBOL Source          ICFLIB/CTDMUL          RCHAS400 10/03/90 14:34:16          Page 2
STMT SEQNBR -A 1 B..+...2....+...3....+...4....+...5....+...6....+...7..IDENTFCN S COPYNAME CHG DATE
 1 000100 IDENTIFICATION DIVISION.                                10/01/87
 2 000200 PROGRAM-ID. CTFMUL.                                     10/01/87
 000300*****
 000400* THIS PROGRAM WILL HANDLE THE REQUEST FOR EITHER A CUSTOMER * 10/01/87
 000500* NUMBER OR AN ITEM NUMBER. THIS IS ACCOMPLISHED BY MAKING * 10/01/87
 000600* THE DATABASE FILE STRUCTURE (KEY LENGTH, KEY POSITION, RECORD * 10/01/87
 000700* LENGTH, RECORD SIZE, ETC.) THE SAME FOR BOTH FILES WITH ONLY * 10/01/87
 000800* THE RECORD CONTENTS DIFFERENT. * 10/01/87
 000900* * 10/01/87
 001000* THIS PROGRAM ENDS WHEN A DETACH REQUEST IS RECEIVED FROM * 10/01/87
 001100* THE SOURCE PROGRAM. * 10/01/87
 001200* * 10/01/87
 001300* INDICATORS ASSOCIATED WITH THE ICF FILE I/O OPERATION * 10/03/90
 001400* ARE DECLARED IN THE WORKING-STORAGE SECTION AND ARE REFERENCED * 10/15/87
 001500* FOR EVERY I/O OPERATION ISSUED. * 10/15/87
 001600*****
 3 001700 ENVIRONMENT DIVISION.                                10/01/87
 4 001800 CONFIGURATION SECTION.                                10/01/87
 5 001900 SOURCE-COMPUTER. IBM-AS400.                            01/15/88
 6 002000 OBJECT-COMPUTER. IBM-AS400.                            01/15/88
 7 002100 SPECIAL-NAMES. I-O-FEEDBACK IS I0-FBA                10/01/87
 8 002200 OPEN-FEEDBACK IS OPEN-FBA.                            10/01/87
 9 002300 INPUT-OUTPUT SECTION.                                  10/01/87
```

Figure 10-26 (Part 1 of 5). Target Program Example — CTFMUL (User-Defined Formats)

```

002400* 1 10/01/87
10 002500 FILE-CONTROL. 10/01/87
11 002600 SELECT PFILE ASSIGN TO DATABASE-PFILE 10/01/87
12 002700 ORGANIZATION IS INDEXED 10/01/87
13 002800 ACCESS IS RANDOM 10/01/87
14 002900 RECORD KEY IS EXTERNALLY-DESCRIBED-KEY 10/01/87
15 003000 WITH DUPLICATES. 10/01/87
16 003100 SELECT CFILF ASSIGN TO WORKSTATION-CFILE-SI 10/01/87
17 003200 ORGANIZATION IS TRANSACTION 10/01/87
18 003300 FILE STATUS IS STATUS-IND MAJ-MIN. 10/01/87
19 003400 SELECT QPRINT ASSIGN TO PRINTER-QSYSPT. 10/01/87
20 003500 DATA DIVISION. 10/01/87
21 003600 FILE SECTION. 10/01/87
22 003700 FD PFILE 10/01/87
23 003800 LABEL RECORDS ARE STANDARD. 10/01/87
24 003900 01 PREC. 10/01/87
25 004000 COPY DDS-ALL-FORMATS OF PFILE. 10/01/87
26 +000001 05 PFILE-RECORD PIC X(150). <-ALL-FMTS
+000002* I-O FORMAT:DBREC FROM FILE PFILE OF LIBRARY ICFLIB <-ALL-FMTS
+000003* <-ALL-FMTS
+000004*THE KEY DEFINITIONS FOR RECORD FORMAT DBREC <-ALL-FMTS
+000005* NUMBER NAME RETRIEVAL TYPE ALTSEQ <-ALL-FMTS
+000006* 0001 DBSEQ ASCENDING AN NO <-ALL-FMTS
27 +000007 05 DBREC REDEFINES PFILE-RECORD. <-ALL-FMTS
28 +000008 06 RECCUS PIC X(1). <-ALL-FMTS
29 +000009 06 DBSEQ PIC X(6). <-ALL-FMTS
30 +000010 06 DBDATA PIC X(130). <-ALL-FMTS
31 +000011 06 DBFILL PIC X(13). <-ALL-FMTS
32 004100 FD CFILE 10/01/87
33 004200 LABEL RECORDS ARE STANDARD. 10/01/87
34 004300 01 ICFREC. 10/01/87
35 004400 COPY DDS-ALL-FORMATS-I-O OF CFILE. 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTDMUL RCHAS400 10/03/90 14:34:16 Page 3
STMT SEQNBR -A 1 B...2...3...4...5...6...7...IDENTFCN S COPYNAME CHG DATE
36 +000001 05 CFILE-RECORD PIC X(150). <-ALL-FMTS
+000002* INPUT FORMAT:SNDPART FROM FILE CFILE OF LIBRARY ICFLIB <-ALL-FMTS
+000003* <-ALL-FMTS
37 +000004 05 SNDPART-I REDEFINES CFILE-RECORD. <-ALL-FMTS
38 +000005 06 RECTYP PIC X(1). <-ALL-FMTS
39 +000006 06 ITEMNO PIC X(6). <-ALL-FMTS
40 +000007 06 EDATA PIC X(130). <-ALL-FMTS
41 +000008 06 FILL1 PIC X(13). <-ALL-FMTS
+000009* OUTPUT FORMAT:SNDPART FROM FILE CFILE OF LIBRARY ICFLIB <-ALL-FMTS
+000010* <-ALL-FMTS
42 +000011 05 SNDPART-O REDEFINES CFILE-RECORD. <-ALL-FMTS
43 +000012 06 RECTYP PIC X(1). <-ALL-FMTS
44 +000013 06 ITEMNO PIC X(6). <-ALL-FMTS
45 +000014 06 EDATA PIC X(130). <-ALL-FMTS
46 +000015 06 FILL1 PIC X(13). <-ALL-FMTS
+000016* INPUT FORMAT:RCVPART FROM FILE CFILE OF LIBRARY ICFLIB <-ALL-FMTS
+000017* <-ALL-FMTS
47 +000018 05 RCVPART-I REDEFINES CFILE-RECORD. <-ALL-FMTS
48 +000019 06 RECID2 PIC X(6). <-ALL-FMTS
49 +000020 06 PARTDS PIC X(80). <-ALL-FMTS
50 +000021 06 FILL4 PIC X(64). <-ALL-FMTS
+000022* OUTPUT FORMAT:RCVPART FROM FILE CFILE OF LIBRARY ICFLIB <-ALL-FMTS
+000023* <-ALL-FMTS
51 +000024 05 RCVPART-O REDEFINES CFILE-RECORD. <-ALL-FMTS
52 +000025 06 RECID2 PIC X(6). <-ALL-FMTS
53 +000026 06 PARTDS PIC X(80). <-ALL-FMTS
54 +000027 06 FILL4 PIC X(64). <-ALL-FMTS
+000028* INPUT FORMAT:RCVTRND FROM FILE CFILE OF LIBRARY ICFLIB <-ALL-FMTS
+000029* <-ALL-FMTS
+000030* 05 RCVTRND-I REDEFINES CFILE-RECORD. <-ALL-FMTS
+000031* OUTPUT FORMAT:RCVTRND FROM FILE CFILE OF LIBRARY ICFLIB <-ALL-FMTS
+000032* <-ALL-FMTS
+000033* 05 RCVTRND-O REDEFINES CFILE-RECORD. <-ALL-FMTS
55 004500 FD QPRINT 10/01/87
56 004600 LABEL RECORDS ARE OMITTED. 10/01/87
57 004700 01 PRINTREC. 01/14/88
58 004800 05 RC PIC 9999. 01/15/88
59 004900 05 ERRMSG PIC X(128). 01/14/88
60 005000 WORKING-STORAGE SECTION. 10/01/87
61 005100 77 MAJ-MIN-SAV PIC X(4). 10/01/87
62 005200 77 STATUS-IND PIC X(2). 10/01/87
63 005300 77 INDON PIC 1 VALUE B"1". 10/01/87
64 005400 77 INDOFF PIC 1 VALUE B"0". 10/01/87
65 005500 77 LEN PIC 9(10)V9(5) COMP 10/01/87
66 005600 VALUE 0. 10/01/87
67 005700 77 CMD2 PIC X(31) 10/01/87

```

Figure 10-26 (Part 2 of 5). Target Program Example — CTDMUL (User-Defined Formats)


```

68 005800 VALUE "CPYF HEXDUMP *LIST PRFTMT(*HEX)". 10/01/87
69 005900 01 CMNF-INDIC-AREA. 10/01/87
006000* ALLOW WRITE (ALWWRT) INDICATOR 10/01/87
70 006100 05 IN10 PIC 1 INDIC 10. 10/01/87
71 006200 88 IN10-ON VALUE B"1". 10/01/87
72 006300 88 IN10-OFF VALUE B"0". 10/01/87
006400* RECEIVE TURNAROUND (RCVTRNRND) INDICATOR 10/01/87
73 006500 05 IN40 PIC 1 INDIC 40. 10/01/87
74 006600 88 IN40-ON VALUE B"1". 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTDMUL RCHAS400 10/03/90 14:34:16 Page 4
STMT SEQNBR -A 1 B..+....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
75 006700 88 IN40-OFF VALUE B"0". 10/01/87
006800* RECEIVE DETACH (RCVDETACH) INDICATOR 10/01/87
76 006900 05 IN44 PIC 1 INDIC 44. 10/01/87
77 007000 88 IN44-ON VALUE B"1". 10/01/87
78 007100 88 IN44-OFF VALUE B"0". 10/01/87
79 007200 01 MAJ-MIN. 10/01/87
80 007300 05 MAJ PIC X(2). 10/01/87
81 007400 05 MIN PIC X(2). 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTDMUL RCHAS400 10/03/90 14:34:16 Page 5
STMT SEQNBR -A 1 B..+....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
007500/ 10/01/87
82 007600 PROCEDURE DIVISION. 10/01/87
007700 DECLARATIVES. 10/01/87
007800 ERR-SECTION SECTION. 10/01/87
007900***** 10/01/87
008000* 2 10/01/87
008100* 10/01/87
008200 USE AFTER STANDARD ERROR PROCEDURE ON CFILE. 10/01/87
008300 CFILE-EXCEPTION. 10/01/87
008400* 10/01/87
008500* CHECK THE MAJOR/MINOR CODES AND TAKE APPROPRIATE ACTION 10/01/87
008600* 10/01/87
008700* MAJOR CODE 34 - INPUT EXCEPTION. 10/01/87
83 008800 IF MAJ = "34" 10/01/87
008900* DATA TRUNCATED IN INPUT AREA. 10/01/87
84 009000 IF MIN = "31" 10/01/87
85 009100 MOVE MAJ-MIN TO MAJ-MIN-SAV 10/12/87
86 009200 GO TO EXIT-DECLARATIVES 10/12/87
009300 ELSE 10/12/87
87 009400 GO TO EXIT-DECLARATIVES. 10/12/87
009500***** 10/01/87
009600* * 02/21/89
009700* PRINT A MESSAGE SAYING CTD MUL PROGRAM ENDED ABNORMALLY. * 02/21/89
009800* CLOSE ALL THE FILES AND END THE PROGRAM. THIS ROUTINE IS CALLED * 02/21/89
009900* WHEN A NON-RECOVERABLE ERROR OCCURS IN ICF FILE. * 10/03/90
010000* * 02/21/89
010100***** 10/01/87
010200 GETFBA. 10/01/87
88 010300 MOVE MAJ-MIN TO RC. 01/14/88
89 010400 MOVE "CTDMUL HAS COMPLETED ABNORMALLY" TO ERRMSG. 01/14/88
90 010500 WRITE PRINTREC. 10/01/87
91 010600 CLOSE PFILE 10/01/87
010700 CFILE 10/01/87
010800 QPRINT. 10/01/87
92 010900 STOP RUN. 10/01/87
011000* 10/01/87
011100 EXIT-DECLARATIVES. 10/01/87
011200 EXIT. 10/01/87
011300* 10/01/87
93 011400 END DECLARATIVES. 10/01/87
011500***** 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTDMUL RCHAS400 10/03/90 14:34:16 Page 6
STMT SEQNBR -A 1 B..+....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME CHG DATE
011600/ 10/01/87
011700 START-PROGRAM SECTION. 10/01/87
011800* 02/27/89
011900 START-PROGRAM-PARAGRAPH. 10/01/87
012000* 3 10/01/87
94 012100 OPEN OUTPUT QPRINT 10/01/87
012200 I-O CFILE 10/01/87
012300 INPUT PFILE. 10/01/87
012400***** 10/01/87
012500* * 10/01/87
012600* READ THE REQUEST FROM THE SOURCE PROGRAM. INDICATOR 40 * 10/01/87
012700* INDICATES RCVTRNRND OCCURRED. INDICATOR 44 INDICATES THAT * 10/01/87
012800* DETACH INDICATOR HAS BEEN RECEIVED FROM THE REMOTE SYSTEM. * 10/01/87
012900* * 10/01/87
013000* THIS PROGRAM CHECKS FOR ERRORS ON EVERY ICF FILE * 10/03/90
013100* FILE OPERATION. A MAJOR CODE GREATER THAN 03 INDICATES * 10/01/87

```

Figure 10-26 (Part 3 of 5). Target Program Example — CTD MUL (User-Defined Formats)

```

013200* AN ERROR. * 10/01/87
013300* * 10/01/87
013400***** 10/01/87
013500* 4 10/01/87
013600 RECEIVE-DATA. 10/01/87
95 013700 PERFORM READ-CFILE THRU READ-CFILE-EXIT. 10/01/87
96 013800 IF IN40-ON 10/01/87
97 013900 GO TO SEND-DATA. 10/01/87
98 014000 PERFORM RCVTRNRND THRU RCVTRNRND-EXIT 10/01/87
014100 UNTIL IN40-ON. 10/01/87
014200***** 10/01/87
014300* * 10/01/87
014400* A REQUEST FROM THE SOURCE PROGRAM RESULTS IN READING A SINGLE * 10/01/87
014500* RECORD CONTAINING THE REQUESTED CUSTOMER OR ORDER NUMBER. THE * 10/01/87
014600* RESPONSE WILL BE RETURNED IN A SINGLE RECORD CONTAINING EITHER * 10/01/87
014700* THE ITEM OR CUSTOMER INFORMATION, DEPENDING ON THE DATA BASE * 10/01/87
014800* CONTENT. * 10/01/87
014900* * 10/01/87
015000* THE RESPONSE IS SENT TO THE SOURCE PROGRAM BY WRITING TO THE * 10/01/87
015100* PROGRAM DEVICE FILE USING FORMAT SNDPART. * 10/15/87
015200* * 10/01/87
015300* WHEN THE REQUESTED CUSTOMER OR ITEM NUMBER IS NOT FOUND, * 10/01/87
015400* 000000 IS PROPAGATED TO THE KEY FIELD BEFORE THE RESPONSE * 10/01/87
015500* IS SENT BACK TO THE SOURCE PROGRAM. * 10/01/87
015600* * 02/22/89
015700***** 10/01/87
015800* 10/01/87
015900* 5 10/01/87
016000 SEND-DATA. 10/01/87
99 016100 MOVE RECID2 OF RCVPART-I TO DBSEQ. 10/01/87
100 016200 READ PFILE INVALID KEY MOVE 0 TO DBSEQ. 10/01/87
102 016300 MOVE RECCUS TO RECTYP OF SNDPART-0. 10/01/87
103 016400 MOVE ZEROS TO CMNF-INDIC-AREA. 10/01/87
104 016500 MOVE DBSEQ TO ITEMNO OF SNDPART-0. 10/01/87
105 016600 MOVE DBDATA TO EDATA OF SNDPART-0 10/01/87
106 016700 WRITE ICFREC FROM PREC FORMAT IS "SNDPART" 10/01/87
016800 INDICATORS ARE CMNF-INDIC-AREA. 10/01/87
107 016900 GO TO RECEIVE-DATA. 10/01/87
017000***** 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTDMUL RCHAS400 10/03/90 14:34:16 Page 7
STMT SEQNBR -A 1 B.....2.....3.....4.....5.....6.....7..IDENTFCN S COPYNAME CHG DATE
017100* * 10/01/87
017200* THIS ROUTINE ISSUES READ OPERATION TO THE PROGRAM DEVICE. * 10/15/87
017300* DETACH INDICATION IS CHECKED FOR AND IF IT OCCURRED, THE * 10/01/87
017400* PROGRAM IS ENDED (IN44-ON). * 10/01/87
017500* * 10/01/87
017600***** 10/01/87
017700* 6 10/01/87
017800 READ-CFILE. 10/01/87
108 017900 MOVE ZEROS TO CMNF-INDIC-AREA. 10/01/87
109 018000 READ CFILE FORMAT IS "RCVPART" 10/01/87
018100 INDICATORS ARE CMNF-INDIC-AREA. 10/01/87
110 018200 SET IN40-ON TO TRUE. 01/25/88
111 018300 IF IN44-ON 10/01/87
112 018400 GO TO END-PROGRAM. 10/01/87
018500 READ-CFILE-EXIT. 10/01/87
018600 EXIT. 02/28/89
018700* 10/01/87
018800***** 10/01/87
018900* * 02/21/89
019000* THIS ROUTINE READS THE ICF FILE UNTIL RCVTRNRND OCCURS. * 10/03/90
019100* DETACH INDICATION IS CHECKED FOR AND IF IT OCCURRED, THE * 10/01/87
019200* PROGRAM IS ENDED (IN44-ON). * 10/01/87
019300* * 02/21/89
019400***** 10/01/87
019500* 7 10/01/87
113 019600 RCVTRNRND. 10/01/87
114 019700 MOVE ZEROS TO CMNF-INDIC-AREA. 10/01/87
115 019800 READ CFILE FORMAT IS "RCVTRND" 10/01/87
019900 INDICATORS ARE CMNF-INDIC-AREA. 10/01/87
116 020000 IF IN44-ON 10/01/87
117 020100 GO TO END-PROGRAM. 10/01/87
020200 RCVTRNRND-EXIT. 10/01/87
020300 EXIT. 02/28/89
020400* 10/01/87
020500***** 10/01/87
020600* * 02/21/89
020700* ROUTINE TO END THE JOB AND CLOSE THE FILES. * 02/21/89
020800* * 02/21/89
020900***** 10/01/87

```

Figure 10-26 (Part 4 of 5). Target Program Example — CTDMUL (User-Defined Formats)

```

021000*                                     10/01/87
021100* 6                                     10/14/87
118 021200 END-PROGRAM.                     10/01/87
119 021300 MOVE MAJ-MIN TO RC.              01/14/88
120 021400 MOVE "CTDMUL HAS COMPLETED NORMALLY" TO ERRMSG. 01/14/88
121 021500 WRITE PRINTREC.                  10/01/87
122 021600 CLOSE PFILE                      10/01/87
    021700 CFILE                             10/01/87
    021800 OPRINT.                            10/01/87
123 021900 STOP RUN.                        10/01/87
***** END OF SOURCE *****
5738CB1 V2R1M0 910524 AS/400 COBOL Messages ICFLIB/CTDMUL RCHAS400 10/03/90 14:34:16 Page 8
STMT
* 35 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 004400
Message . . . . : No INPUT fields found for format RCVTRND.
* 35 MSGID: LBL0600 SEVERITY: 10 SEQNBR: 004400
Message . . . . : No OUTPUT fields found for format RCVTRND.
***** END OF MESSAGES *****
Message Summary
Total   Info(0-4)   Warning(5-19)   Error(20-29)   Severe(30-39)   Terminal(40-99)
  2         0           2             0             0             0
Source records read . . . . . : 219
Copy records read . . . . . : 44
Copy members processed . . . . . : 2
Sequence errors . . . . . : 0
Highest severity message issued . . : 10
LBL0901 00 Program CTDML created in library ICFLIB.
***** END OF COMPILATION *****

```

Figure 10-26 (Part 5 of 5). Target Program Example — CTDML (User-Defined Formats)

```

5738CB1 V2R1M0 910524          IBM AS/400 COBOL/400          ICFLIB/CTFMUL          RCHAS400 10/03/90 14:38:04          Page 1
Program . . . . . : CTFMUL
Library . . . . . : ICFLIB
Source file . . . . . : QICFPUB
Library . . . . . : ICFLIB
Source member . . . . . : CTFMUL 10/03/90 14:32:29
Generation severity level . . . . . : 29
Text 'description' . . . . . : CBL Multiple Session Inquiry - Target $$
Source listing options . . . . . : *SOURCE
Generation options . . . . . : *NONE
Message limit:
Number of messages . . . . . : *NOMAX
Message limit severity . . . . . : 29
Print file . . . . . : QSYSVRT
Library . . . . . : *LIBL
FIPS flagging . . . . . : *NOFIPS *NOSEG *NODEB *NOOBSOLETE
SAA flagging . . . . . : *NOFLAG
Flagging severity . . . . . : 0
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Compiler . . . . . : IBM AS/400 COBOL/400
5738CB1 V2R1M0 910524          AS/400 COBOL Source          ICFLIB/CTFMUL          RCHAS400 10/03/90 14:38:04          Page 2

```

```

STMT SEQNBR -A 1 B.+. . . . . 2. . . . . 3. . . . . 4. . . . . 5. . . . . 6. . . . . 7. IDENTFCN S COPYNAME CHG DATE
1 000100 IDENTIFICATION DIVISION. 10/01/87
2 000200 PROGRAM-ID. CTFMUL. 10/01/87
000300***** 10/01/87
000400* THIS PROGRAM WILL HANDLE THE REQUEST FOR EITHER A CUSTOMER * 10/01/87
000500* NUMBER OR AN ITEM NUMBER. THIS IS ACCOMPLISHED BY MAKING * 10/01/87
000600* THE DATABASE FILE STRUCTURE (KEY LENGTH, KEY POSITION, RECORD * 10/01/87
000700* LENGTH, RECORD SIZE, ETC.) THE SAME FOR BOTH FILES WITH ONLY * 10/01/87
000800* THE RECORD CONTENTS DIFFERENT. * 10/01/87
000900* * 10/01/87
001000* THIS PROGRAM ENDS WHEN A DETACH REQUEST IS RECEIVED FROM * 10/01/87
001100* THE SOURCE PROGRAM. * 10/01/87
001200* * 10/01/87
001300* THIS PROGRAM USES THE SYSTEM-SUPPLIED FORMAT TO ISSUE THE I/O * 10/01/87
001400* OPERATION AND THEREFORE, DOES NOT USE THE OPTION INDICATORS * 10/01/87
001500* ASSOCIATED WITH THE KEYWORDS. NOTICE THAT THE ICF FILE * 10/03/90
001600* FILE DECLARATION SELECT STATEMENT REFLECTS THE USE OF A * 10/15/87
001700* SEPARATE INDICATOR AREA FOR INDICATORS. * 10/15/87
001800***** 10/01/87
3 001900 ENVIRONMENT DIVISION. 10/01/87
4 002000 CONFIGURATION SECTION. 10/01/87
5 002100 SOURCE-COMPUTER. IBM-AS400. 01/15/88
6 002200 OBJECT-COMPUTER. IBM-AS400. 01/15/88
7 002300 SPECIAL-NAMES. I-O-FEEDBACK IS IO-FBA 10/01/87
8 002400 OPEN-FEEDBACK IS OPEN-FBA. 10/01/87
9 002500 INPUT-OUTPUT SECTION. 10/01/87
002600* 1 10/01/87
10 002700 FILE-CONTROL. 10/01/87
11 002800 SELECT PFILE ASSIGN TO DATABASE-PFILE 10/01/87
12 002900 ORGANIZATION IS INDEXED 10/01/87
13 003000 ACCESS IS RANDOM 10/01/87
14 003100 RECORD KEY IS EXTERNALLY-DESCRIBED-KEY 10/01/87
15 003200 WITH DUPLICATES. 10/01/87
16 003300 SELECT CFIL ASSIGN TO WORKSTATION-CFILE-SI 10/01/87
17 003400 ORGANIZATION IS TRANSACTION 10/01/87
18 003500 FILE STATUS IS STATUS-IND MAJ-MIN. 10/01/87
19 003600 SELECT QPRINT ASSIGN TO PRINTER-QSYSVRT. 10/01/87
20 003700 DATA DIVISION. 10/01/87
21 003800 FILE SECTION. 10/01/87
22 003900 FD PFILE 10/01/87
23 004000 LABEL RECORDS ARE STANDARD. 10/01/87
24 004100 01 PREC. 10/01/87
25 004200 COPY DDS-ALL-FORMATS OF PFILE. 10/01/87
26 +000001 05 PFILE-RECORD PIC X(150). <-ALL-FMTS
+000002* I-O FORMAT:DBREC FROM FILE PFILE OF LIBRARY ICFLIB <-ALL-FMTS
+000003* <-ALL-FMTS
+000004*THE KEY DEFINITIONS FOR RECORD FORMAT DBREC <-ALL-FMTS
+000005* NUMBER NAME RETRIEVAL TYPE ALTSEQ <-ALL-FMTS
+000006* 0001 DBSEQ ASCENDING AN NO <-ALL-FMTS
27 +000007 05 DBREC REDEFINES PFILE-RECORD. <-ALL-FMTS
28 +000008 06 RECCUS PIC X(1). <-ALL-FMTS
29 +000009 06 DBSEQ PIC X(6). <-ALL-FMTS
30 +000010 06 DBDATA PIC X(130). <-ALL-FMTS
31 +000011 06 DBFILL PIC X(13). <-ALL-FMTS
32 004300 FD CFILE 10/01/87
33 004400 LABEL RECORDS ARE STANDARD. 10/01/87

```

Figure 10-27 (Part 1 of 4). Target Program Example — CTFMUL (System-Supplied Formats)

```

5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTFMUL RCHAS400 10/03/90 14:38:04 Page 3
STMT SEQNBR -A 1 B.+...2...+...3...+...4...+...5...+...6...+...7...IDENTFCN S COPYNAME CHG DATE
34 004500 01 ICFREC. 10/01/87
35 004600 05 SNDPART. 10/01/87
36 004700 10 LNGTH PIC 9(4). 02/22/89
37 004800 10 RECTYP PIC X. 10/01/87
38 004900 10 ITEMNO PIC X(6). 10/01/87
39 005000 10 EDATA PIC X(130). 10/01/87
40 005100 10 FILL1 PIC X(13). 10/01/87
41 005200 05 RCVPART REDEFINES SNDPART. 10/01/87
42 005300 10 RECID2 PIC 9(6). 10/01/87
43 005400 10 PARTDS PIC X(80). 10/01/87
44 005500 10 FILL4 PIC X(64). 10/01/87
45 005600 FD QPRINT 10/01/87
46 005700 LABEL RECORDS ARE OMITTED. 10/01/87
47 005800 01 PRINTREC. 01/14/88
48 005900 05 RC PIC 9999. 01/14/88
49 006000 05 ERRMSG PIC X(128). 01/14/88
50 006100 WORKING-STORAGE SECTION. 10/01/87
51 006200 77 MAJ-MIN-SAV PIC X(4). 10/01/87
52 006300 77 STATUS-IND PIC X(2). 10/01/87
53 006400 77 INDON PIC 1 VALUE B"1". 10/01/87
54 006500 77 INDOFF PIC 1 VALUE B"0". 10/01/87
55 006600 77 LEN PIC 9(10)V9(5) COMP 10/01/87
56 006700 VALUE 0. 10/01/87
57 006800 77 CMD2 PIC X(31) 10/01/87
58 006900 VALUE "CPYF HEXDUMP *LIST PRFTMT(*HEX)". 10/01/87
007000* 10/08/87
59 007100 01 CMNF-INDIC-AREA. 10/01/87
007200* RECEIVE TURNAROUND (RCVTRNRND) INDICATOR 10/01/87
60 007300 05 IN40 PIC 1 INDIC 40. 10/01/87
61 007400 88 IN40-ON VALUE B"1". 10/01/87
62 007500 88 IN40-OFF VALUE B"0". 10/01/87
007600* RECEIVE DETACH (RCVDETACH) INDICATOR 10/01/87
63 007700 05 IN44 PIC 1 INDIC 44. 10/01/87
64 007800 88 IN44-ON VALUE B"1". 10/01/87
65 007900 88 IN44-OFF VALUE B"0". 10/01/87
008000* 10/08/87
66 008100 01 MAJ-MIN. 10/01/87
67 008200 05 MAJ PIC X(2). 10/01/87
68 008300 05 MIN PIC X(2). 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTFMUL RCHAS400 10/03/90 14:38:04 Page 4
STMT SEQNBR -A 1 B.+...2...+...3...+...4...+...5...+...6...+...7...IDENTFCN S COPYNAME CHG DATE
008400/ 10/01/87
69 008500 PROCEDURE DIVISION. 10/01/87
008600 DECLARATIVES. 10/01/87
008700 ERR-SECTION SECTION. 10/01/87
008800***** 10/01/87
008900* 2 10/01/87
009000* 10/01/87
009100 USE AFTER STANDARD ERROR PROCEDURE ON CFILE. 10/01/87
009200 CFILE-EXCEPTION. 10/01/87
009300* 10/01/87
009400* CHECK THE MAJOR/MINOR CODES AND TAKE APPROPRIATE ACTION 10/01/87
009500* MAJOR CODE 34 - INPUT EXCEPTION. 10/08/87
009600* 10/01/87
70 009700 IF MAJ = "34" 10/01/87
009800* DATA TRUNCATED IN INPUT AREA. 10/01/87
71 009900 IF MIN = "31" 10/01/87
72 010000 MOVE MAJ-MIN TO MAJ-MIN-SAV 10/10/87
73 010100 GO TO EXIT-DECLARATIVES 10/10/87
010200 ELSE 10/10/87
74 010300 GO TO EXIT-DECLARATIVES. 10/10/87
010400* 10/08/87
010500***** 10/01/87
010600* * 02/21/89
010700* PRINT A MESSAGE SAYING CTFMUL PROGRAM ENDED ABNORMALLY. * 02/21/89
010800* CLOSE ALL THE FILES AND END THE PROGRAM. THIS ROUTINE IS CALLED * 02/21/89
010900* WHEN A NON-RECOVERABLE ERROR OCCURS IN THE ICF FILE. * 10/03/90
011000* * 02/21/89
011100***** 10/01/87
011200* 10/08/87
011300 GETFBA. 10/01/87
75 011400 MOVE MAJ-MIN TO RC. 01/14/88
76 011500 MOVE "CTFMUL HAS COMPLETED ABNORMALLY" TO ERRMSG. 01/14/88
77 011600 WRITE PRINTREC. 10/01/87
78 011700 CLOSE PFILE 10/01/87
011800 CFILE 10/01/87
011900 QPRINT. 10/01/87
79 012000 STOP RUN. 10/01/87

```

Figure 10-27 (Part 2 of 4). Target Program Example — CTFMUL (System-Supplied Formats)

```

012100* 10/01/87
012200 EXIT-DECLARATIVES. 10/01/87
012300 EXIT. 10/01/87
012400* 10/01/87
80 012500 END DECLARATIVES. 10/01/87
012600***** 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTFMUL RCHAS400 10/03/90 14:38:04 Page 5
STMT SEQNBR -A 1 B.+...2.+...3.+...4.+...5.+...6.+...7..IDENTFCN S COPYNAME CHG DATE
012700/ 10/01/87
012800 START-PROGRAM SECTION. 10/01/87
012900 START-PROGRAM-PARAGRAPH. 10/01/87
013000* 3 10/01/87
81 013100 OPEN OUTPUT OPRINT 10/01/87
013200 I-O CFILE 10/01/87
013300 INPUT PFILE. 10/01/87
013400***** 10/01/87
013500* * 10/01/87
013600* READ THE REQUEST FROM THE SOURCE PROGRAM. MINOR RETURN CODE '00'* 10/01/87
013700* INDICATES RCVTRNRD OCCURRED. MINOR RETURN CODE OF '08' * 02/21/89
013800* INDICATES DETACH HAS BEEN RECEIVED. * 10/15/87
013900* * 10/01/87
014000* THIS PROGRAM CHECKS FOR ERRORS ON EVERY ICF FILE I/O * 10/03/90
014100* OPERATION. A MAJOR CODE GREATER THAN 03 INDICATES AN ERROR. * 02/22/89
014200* * 10/01/87
014300***** 10/01/87
014400* 4 10/01/87
014500 RECEIVE-DATA. 10/01/87
82 014600 MOVE SPACES TO MAJ-MIN. 10/10/87
83 014700 PERFORM READ-CFILE THRU READ-CFILE-EXIT UNTIL 10/01/87
014800 MIN IS EQUAL TO "00". 10/01/87
014900***** 10/01/87
015000* * 10/01/87
015100* A REQUEST FROM THE SOURCE PROGRAM RESULTS IN READING A SINGLE * 10/01/87
015200* RECORD CONTAINING THE REQUESTED CUSTOMER OR ORDER NUMBER. THE * 10/01/87
015300* RESPONSE WILL BE RETURNED IN A SINGLE RECORD CONTAINING EITHER * 10/01/87
015400* THE ITEM OR CUSTOMER INFORMATION, DEPENDING ON THE DATABASE * 10/01/87
015500* CONTENT. * 10/01/87
015600* * 10/01/87
015700* THE RESPONSE IS SENT TO THE SOURCE PROGRAM BY WRITING TO THE * 10/01/87
015800* PROGRAM DEVICE FILE USING FORMAT SNDPART. * 10/15/87
015900* * 10/01/87
016000* WHEN THE REQUESTED CUSTOMER OR ITEM NUMBER IS NOT FOUND, * 10/01/87
016100* 000000 IS PROPAGATED TO THE KEY FIELD BEFORE THE RESPONSE * 10/01/87
016200* IS SENT BACK TO THE SOURCE PROGRAM. * 10/01/87
016300* * 02/22/89
016400***** 10/01/87
016500* 10/01/87
016600* 5 10/01/87
016700 SEND-DATA. 10/01/87
84 016800 MOVE RECID2 OF RCVPART TO DBSEQ. 10/01/87
85 016900 READ PFILE INVALID KEY MOVE 000000 TO DBSEQ. 10/01/87
87 017000 MOVE RECCUS TO RECTYP. 10/01/87
88 017100 MOVE 150 TO LGTH OF SNDPART. 02/22/89
89 017200 MOVE DBSEQ TO ITEMNO OF SNDPART. 10/01/87
90 017300 MOVE DBDATA TO EDATA OF SNDPART. 10/01/87
91 017400 WRITE ICFREC FORMAT IS "$$SEND". 10/01/87
92 017500 GO TO RECEIVE-DATA. 10/01/87
017600***** 10/01/87
017700* * 02/21/89
017800* THIS ROUTINE ISSUES THE READ OPERATION TO THE PROGRAM DEVICE * 02/21/89
017900* UNTIL RCVTRNRD OCCURS. * 02/21/89
018000* DETACH INDICATION IS CHECKED FOR AND IF IT OCCURRED, THE * 10/01/87
018100* PROGRAM IS ENDED (RC= 08). * 10/01/87
5738CB1 V2R1M0 910524 AS/400 COBOL Source ICFLIB/CTFMUL RCHAS400 10/03/90 14:38:04 Page 6
STMT SEQNBR -A 1 B.+...2.+...3.+...4.+...5.+...6.+...7..IDENTFCN S COPYNAME CHG DATE
018200* * 02/21/89
018300***** 10/15/87
018400* 6 10/01/87
018500 READ-CFILE. 10/01/87
93 018600 READ CFILE 10/01/87
018700 INDICATORS ARE CMNF-INDIC-AREA. 10/01/87
94 018800 IF MIN = "08" 10/01/87
95 018900 GO TO END-PROGRAM. 10/01/87
019000 READ-CFILE-EXIT. 10/01/87
019100 EXIT. 02/28/89
019200* 10/01/87
019300***** 10/01/87
019400* * 02/21/89
019500* ROUTINE TO END THE JOB AND CLOSE THE FILES. * 02/21/89
019600* * 02/21/89

```

Figure 10-27 (Part 3 of 4). Target Program Example — CTFMUL (System-Supplied Formats)

```

019700*****
019800*
019900*(H)
96 020000 END-PROGRAM.
97 020100 MOVE MAJ-MIN TO RC.
98 020200 MOVE "CTFMUL HAS COMPLETED NORMALLY" TO ERRMSG.
99 020300 WRITE PRINTREC.
100 020400 CLOSE PFILE
020500 CFILE
020600 QPRINT.
101 020700 STOP RUN.
          ***** END OF SOURCE *****
5738CB1 V2R1M0 910524 AS/400 COBOL Messages ICFLIB/CTFMUL RCHAS400 10/03/90 14:38:04 Page 7
STMT
          ***** END OF MESSAGES *****
          Message Summary
Total   Info(0-4)   Warning(5-19)   Error(20-29)   Severe(30-39)   Terminal(40-99)
0       0           0               0               0               0
Source records read . . . . . : 207
Copy records read . . . . . : 11
Copy members processed . . . . . : 1
Sequence errors . . . . . : 0
Highest severity message issued . . : 0
LBL0901 00 Program CTFMUL created in library ICFLIB.
          ***** END OF COMPILATION *****

```

Figure 10-27 (Part 4 of 4). Target Program Example — CTFMUL (System-Supplied Formats)

Chapter 11. Communications Applications with RPG/400 Programming Language

Previous chapters in this manual describe the functions provided by ICF. This chapter introduces you to the RPG/400 interface for ICF and provides program examples.

Two program examples are presented in this chapter. For each example, both the source and target programs are provided. Each program is written first with user-defined formats (data description specifications or DDS) and then with system-supplied formats.

The first example is a batch data transfer application using a single session. The second example is a multiple-session inquiry application using one display file and four ICF sessions.

Not all programming considerations or techniques are illustrated in each example in this chapter. Review these examples and the examples provided in the appropriate programming manual before beginning application design and coding.

Note: The examples in this chapter were written to the APPC communications type. Minor changes might be required if another communications type is used.

Introduction to the RPG/400 Interface

Before you write an RPG/400 communications application, you must understand the high-level language interface provided by RPG/400. ICF files are defined as WORKSTN files in RPG/400.

The operations you use in the communications portion of your program are similar to work station operations. In the noncommunications portion of your program, you can use all noncommunications operations you normally use to

process data that is sent or received between your program and the remote program.

Figure 11-1 briefly introduces the RPG/400 statements you use in the communications portion of your program.

Figure 11-1 (Page 1 of 2). RPG/400 Statements

ICF Operation	RPG/400 Operation Code	Function
Open	OPEN	Opens the ICF file
Acquire	ACQ	Acquires a program device to establish a session
Get-attributes	POST ¹	Gets the status information of a program device
Read	READ ²	Receives data from a specific program device
Read-from-invited-program-devices	READ ²	Receives data from any invited program device ³
Write	WRITE	Performs many of the ICF communications functions in a session
Write/Read	EXFMT	Performs the specified function and then receives data from the remote system
Release	REL	Releases the program device to end the session

Figure 11-1 (Page 2 of 2). RPG/400 Statements

ICF Operation	RPG/400 Operation Code	Function
Close	CLOSE	Closes the ICF file
1		The POST operation can retrieve either input/output (I/O) feedback information or the get-attributes. The information you get depends on the factors specified with the POST.
2		An RPG/400 read operation can be directed either to a specific program device or to all invited program devices. The support provided by the RPG/400 compiler determines whether to issue an ICF read or read-from-invited-program-devices operation based on the format of the read operation. For example, if a read is sent with a specific format or terminal specified, the read operation is interpreted as an ICF READ operation. Refer to the RPG/400 language manual for more information.
3		The read-from-invited-program-devices operation could complete without data if the timer interval established with either the timer function or wait record (WAITRCD) ends, or your job is ended (controlled).

Refer to the appropriate RPG/400 manual for details on the syntax and function of each operation.

Error Handling: The INFDS option of the RPG/400 file specification allows you to obtain specific exception or error information for a file by defining and naming a data structure to contain error information, such as the ICF major and minor return codes.

You must understand the relationship between RPG *STATUS values and ICF major/minor return codes.

Figure 11-2 shows the *STATUS values as returned in the RPG/400 INFDS for each major and minor return code. Use this list to determine the ICF return code or group of codes that corresponds to the *STATUS value.

Figure 11-2. *STATUS Values for Major and Minor Return Codes

Major Code	Minor Code	*STATUS Value
00	All	00000
02	All	00000

Figure 11-2. *STATUS Values for Major and Minor Return Codes

Major Code	Minor Code	*STATUS Value
03	All (except 09 and 10)	00000
03	09	01282
03	10	01331
04	All	01299
08	00	01285
11	00	00011
34	All	01201
80, 81	All	01251
82, 83	All	01255

Note: The mapping in Figure 11-2 applies to major/minor codes set as a result of acquire, release, and general I/O operations. Certain major/minor codes are set for open and close errors as well as for other I/O errors. In cases where a major/minor code is set as a result of an open or close error, the return code will map to either the 01205, 01216, or 01217 *STATUS value, depending on which is applicable.

The return code field will not be updated for a *STATUS value of 01285, 01261, or 01281 when an I/O operation was attempted to an unacquired program device, because RPG/400 detects these conditions before calling ICF data management. This mapping is shown in order to note the appropriate RPG/400 *STATUS value to check for the given error condition.

Accessing the Feedback Areas: Use the RPG/400 POST operation to obtain the open or I/O feedback areas for an ICF file. For the RPG/400 support to access information from the I/O feedback area, add the following RPG/400 offset values to the offset values listed in Appendix C.

- 81** Open Feedback Area
- 241** Common I/O Feedback Area
- 367** File Dependent Feedback Area

Example Programs

The programs presented in this section are:

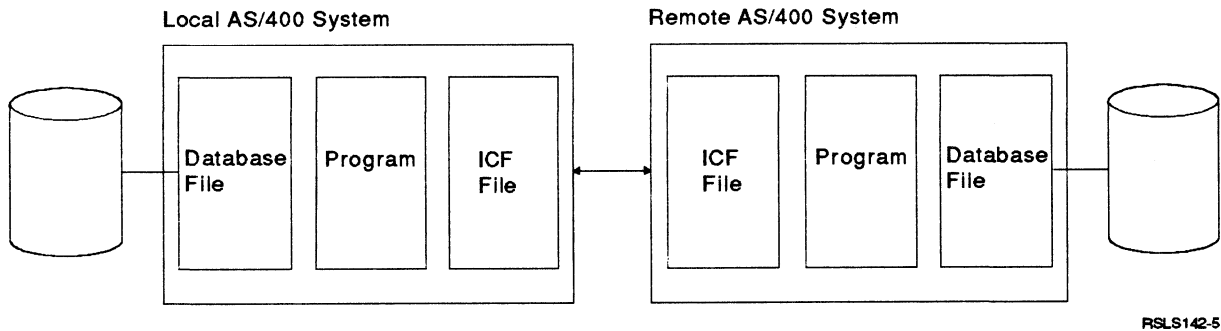
- Example I (Batch Data Transfer)

Figure 11-3 shows a batch data transfer program that reads a database file and sends the data to a remote system. When the source program finishes sending its records, it sends an indication that it is done sending records to the target program. The target program then starts sending its records until it reaches the end-of-file. At end-of-file, the target program sends a detach indication to the source program. The two programs end their sessions.

- Example II (Multiple-Session Inquiry)

Figure 11-4 on page 11-4 shows an inquiry program that accepts inquiries from a display device, sends the request to one of four remote AS/400 systems, and waits for a response to the inquiry. Based on the input received from the display device, the program determines the target program to which it sends the inquiry request. The same program resides in each of the remote systems.

Figure 11-4 on page 11-4 contains a display device and four ICF communications program devices.



RSL5142-5

Figure 11-3. Batch Data Transfer

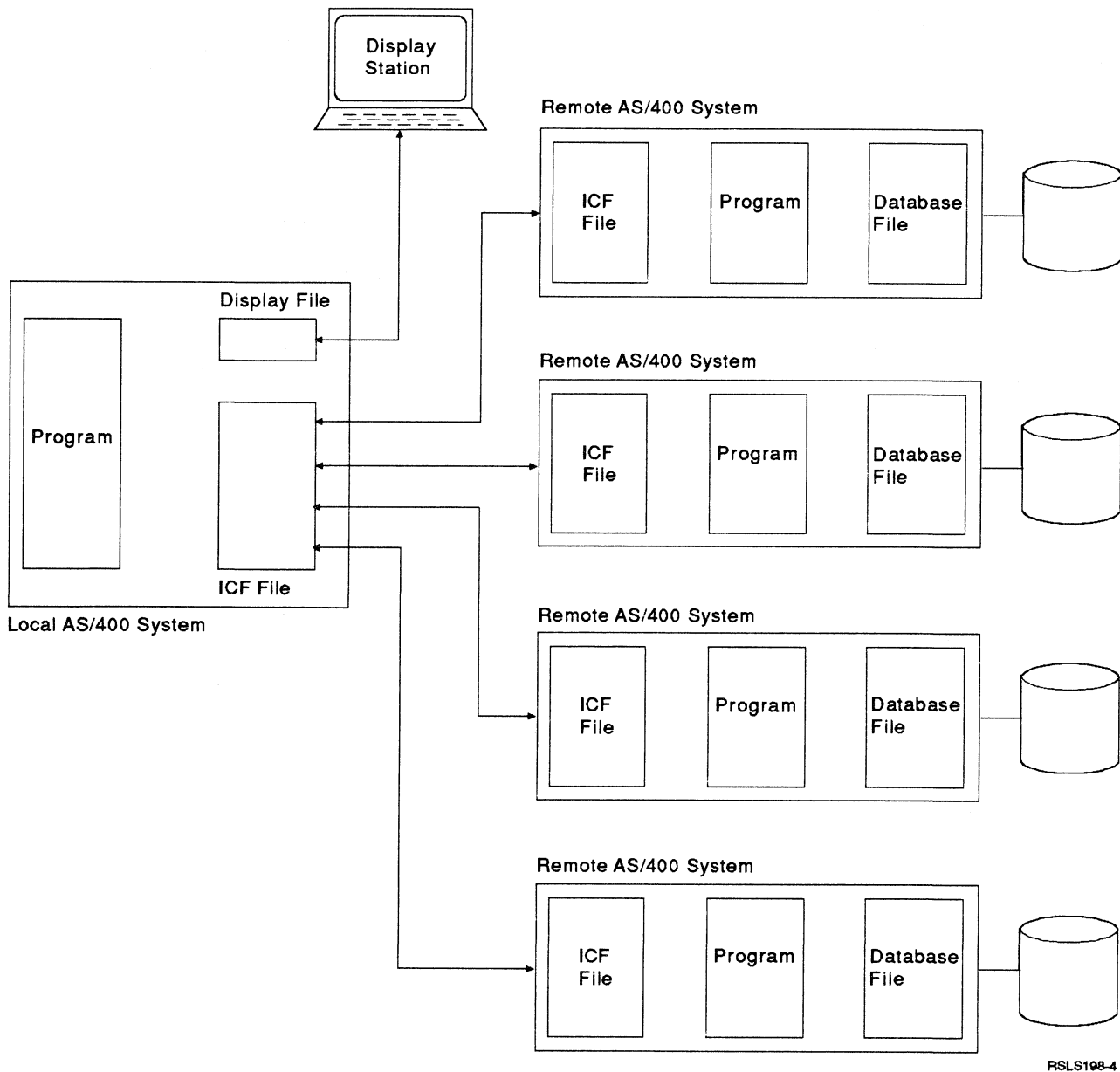


Figure 11-4. Multiple-Session Inquiry

The remainder of this chapter discusses the details of the two application examples. The DDS source for the ICF file, program listings, and an explanation of the programs are included.

Batch Data Transfer (Example I)

The following figures show a batch data transfer program. A source AS/400 system program communicates with a target program on another AS/400 program using the ICF support. The source program starts a target program on a

remote AS/400 system, and transfers a file to that target program.

The target program responds, after receiving an indication that the source is done sending, by reading its own file and then sends the records to the source program until it reaches end-of-file. At end-of-file, the target program sends a detach request to the source program and ends its session.

Both the source program and the target program are described.

Transaction Flow of the Batch Data Transfer (Example I):

In Figure 11-5, the source program issues an evoke to start a program at the remote AS/400 system.

Note: An acquire operation is not necessary since the device was acquired during the open

operation. The device was acquired during the open operation because the ACQPGMDEV parameter was used when the ICF file was created.

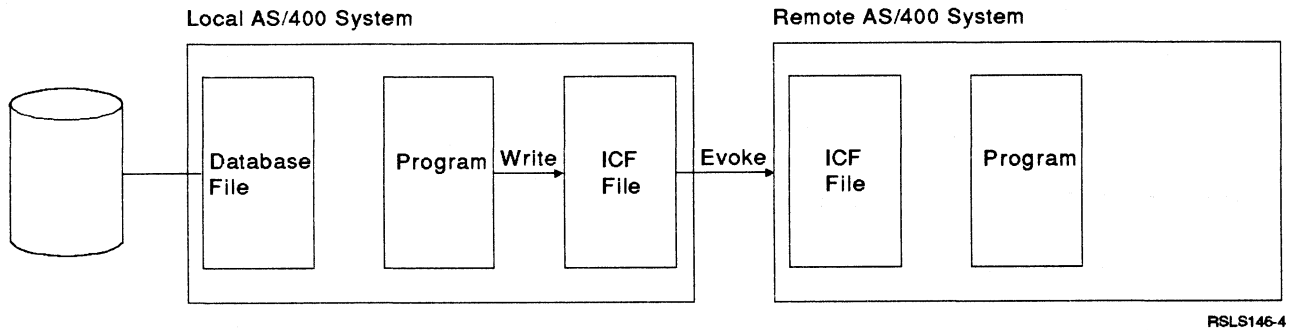


Figure 11-5. Evoke Request Starts a Target Program

After issuing the evoke request, the source program sends a database file to the target

program, which prints the records as shown in Figure 11-6.

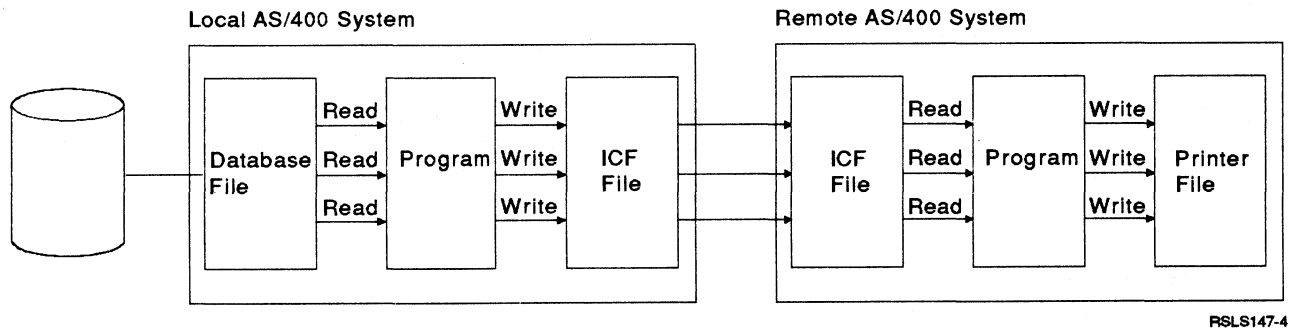


Figure 11-6. Target Program Prints Records

After the target program receives and prints the file, a database file is sent to the source program. The source program prints the

records as they are received as shown in Figure 11-7.

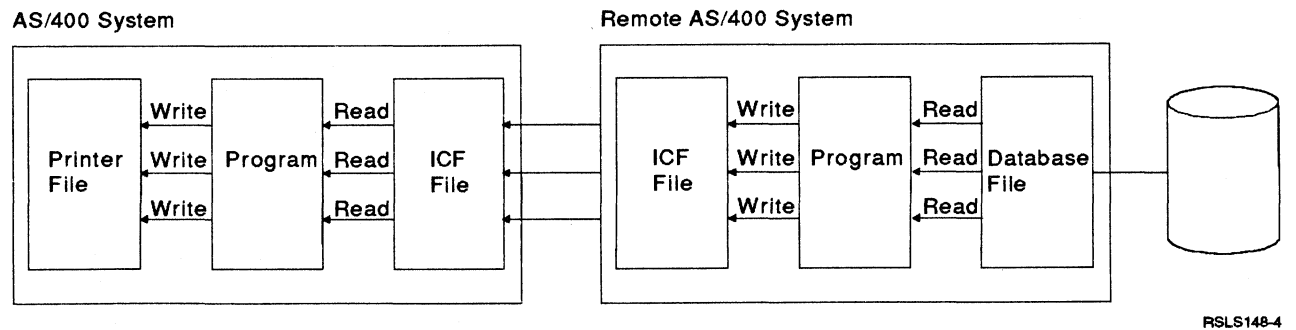
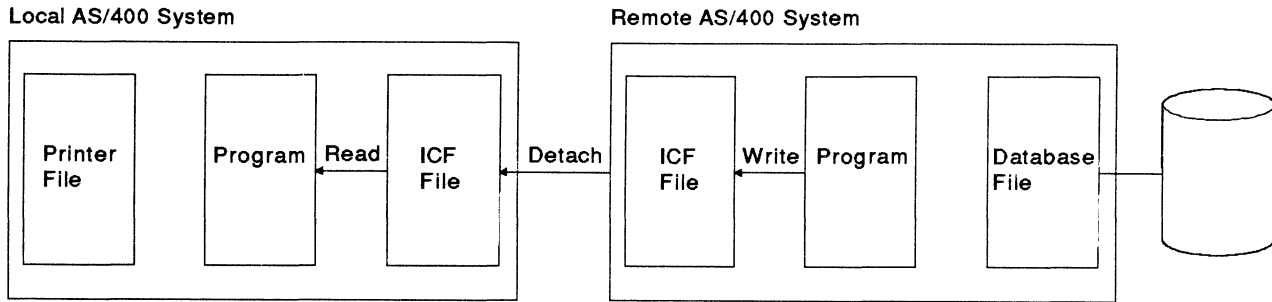


Figure 11-7. Source Program Prints the Received Records

Once all the records have been sent by the target program, the target program issues a

detach to the source program to end the transaction, as shown in Figure 11-8 on page 11-6.



RSLS149-4

Figure 11-8. Target Program Ends the Transaction

Source Program Batch Transfer

(Example 1): The following describes the RPG/400 batch data transfer source program.

Program Files: The RPG/400 batch data transfer source program uses the following files:

SRCICF

An ICF file used to send records to and receive records from the target program.

DBFILE

A database file that contains the records to be sent to the target program.

QPRINT

A printer file that is used to print the records received from the target program.

DDS Source: The DDS used in the ICF file is illustrated below. The other files (DBFILE and QPRINT) are program-described and therefore do not require DDS.

```

A*****
A*                                     *
A*           ICF FILE                 *
A*      USED IN BATCH DATA TRANSFER  *
A*                                     *
A*****
A* FILE LEVEL INDICATORS:
A*
A*           INDARA
A*           RCVTRNRND(15 'END OF DATA')
A*
A 30      DETACH
A*
A*           INDTXT(30 '30->DETACH TARG-
A*           ET PROGRAM.')
A*
A*           RCVDETACH(35 'RECEIVED
A*           DETACHED.')
A*
A*

```

```

A*****
A*                                     *
A*           ICF RECORD FORMATS      *
A*                                     *
A*****
A*           R RCVDATA
A*           RCVFLD          88A
A*           R SNDDATA
A*           SNDFLD          88A
A*           R EVOKPGM
A 58
A 58
A*           EVOKE(&LIB/&PGMID)
A*           SECURITY(2 'PASSWD' +
A*           3 'USERID')
A*           PGMID          10A P
A*           LIB            10A P
A*           R ENDREC
A*           R INVITE
A 45
A*           INVITE

```

ICF File Creation and Program Device Entry

Definition: The command needed to create the ICF file is:

```

CRTICFF FILE(ICFLIB/SRCICF) SRCFILE(ICFLIB/QICFPUB)
SRCMBR(SRCICF) ACQPGMDEV(PGMDEVA) TEXT('ICF FILE
FOR BATCH DATA TRANSFER')

```

The command needed to define the program device entry is:

```

ADDICFDEVE FILE(ICFLIB/SRCICF) PGMDEV(PGMDEVA)
RMTLOCNAME(CHICAGO)

```

Program Explanation: The following describes the structure of the program examples illustrated in Figure 11-9 on page 11-8 and Figure 11-10 on page 11-13. The ICF file used in the first example is defined by the user, and uses externally described data formats (DDS). The second example uses the same file, but uses program-described data and system-supplied formats. The reference numbers in the explanation below correspond to those in the following program examples.

Although the basic structure of the two examples provided is the same, there are differences because of the use of user-defined formats and system-supplied formats. All output operations to the ICF file in the first example are done using the WRITE statement. All output operations to the ICF file in the second example using system-

supplied formats are done using the EXCPT statement.

Differences between the first and second example are described in notes in each of the descriptions.

- 1** The files used in this program are defined in the file specifications section. SRCICF is the ICF file used to send records to the target program.

The files used in the program are opened at the beginning of the RPG/400 cycle and the ICF program device is implicitly acquired because the ACQPGMDEV parameter was specified on the create ICF file (CRTICFF) command.

Note: The input records for SRCICF are explicitly coded in the program using system-supplied formats. because SRCICF is now treated as a program-described file. The system-supplied file QICDMF can be used instead of SRCICF. You can use the system-supplied file by specifying QICDMF in the file specification, or by using an OVRICFF command to change the file name from SRCICF to QICDMF. The OVRICFF command can also be used to change the ACQPGMDEV parameter of the file.

- 2** FEEDBK is the name of the file information data structure (INFDS) used with SRCICF. It contains the following information:
 - Record format name (FMTNM)
 - Program device name (PGMDEV)
 - Major/minor return code (MAJMIN, MAJCOD, MINCOD)

- 3** This section builds and sends the evoke request to the remote system. Because the DDS for the record format only specifies the field identifiers with the record, the program moves the literal value RTDBATCL to the field *PGMID*, and ICFLIB to the field *LIB*. Indicator 50 is set to indicate that the program start request is to be sent.

When the program start request is received at the remote system, ICFLIB is searched for RTDBATCL and that program is then started. RTDBATCL is a control language (CL) program that contains the following:

```
ADDLIB ICFLIB
CALL ICFLIB/RTDBAT
```

Note: In the program using system-supplied formats, the library and program (ICFLIB/RTFBATCL) are specified as part of the \$\$EVOKNI format. RTFBATCL is a CL program that contains the following:

```
ADDLIB ICFLIB
CALL ICFLIB/RTFBAT
```

- 4** This section reads a record from the database file. If the record read is the end-of-file, the program sets indicator 98 on and then goes to **11**.

If it is not the last record, the data is moved to field *SNDFLD* and the program goes to **10** to write the record to the ICF program device. When control returns from **10**, the next database record is read.

- 5** Data is read from the program device associated with the ICF file (SRCICF).
- 6** If an error occurs on the read (return code greater than 03), the error is handled. Otherwise, if data is received (return code not = 03), the data is written to the printer file (QPRINT).

This section reads data records until the detach indication is received from the target program. When the detach is received, indicator 35 is set on, as defined by the RCVD detach keyword in the DDS for the ICF file. Note that RCVD detach is a file level-keyword.

Note: In the program using system-supplied formats, the minor return code of '08' is checked to verify whether the detach is received.

- 7** After the detach request has been received, the following message is written to the print file:

```
RSDBAT HAS COMPLETED NORMALLY
```

The session is ended in **9**.

Note: The program name is RSFBAT in the program using system-supplied formats.

- 8** When an I/O operation to the ICF file (SRCICF) completes unsuccessfully, the following message is written to the print file:

```
RSDBAT HAS COMPLETED ABNORMALLY
```

The session is ended in **9**.

Note: The program name is RSFBAT in the program using system-supplied formats.

9 The program ends the job by setting on last run (LR) indicator and returning to caller of the program. The ICF file is closed, and the session ends at the end of the RPG/400 cycle.

10 This subroutine is called to write data to the program device associated with the ICF file using the format SNDDATA. If an error occurs, the program goes to **3** and a message is printed.

Note: The \$\$\$SENDNI format is used instead of the user-defined SNDDATA

format in the program using system-supplied formats.

11 This subroutine is called to perform an invite request to the ICF program device using format INVITE. If an error occurs, the program goes to **3** and a message is printed.

Note: The \$\$\$SEND format is used instead of the user-defined INVITE format in the program using system-supplied formats.

```

5738R61 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDBAT          10/03/90 14:54:22          Page 1
Compiler . . . . . : IBM AS/400 RPG/400
Command Options:
  Program . . . . . : ICFLIB/RSDBAT
  Source file . . . . . : ICFLIB/QICFPUB
  Source member . . . . . : *PGM
Text not available for message RXT0073 file QRPMSG.
Generation options . . . . . : *NOLIST *NOXREF *NOATR *NODUMP *NOOPTIMIZE
Source listing indentation . . . . . : *NONE
SAA flagging . . . . . : *NOFLAG
Generation severity level . . . . . : 9
Print file . . . . . : *LIBL/QSYSPRT
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Text . . . . . : *SRCMBRTXT
Phase trace . . . . . : *NO
Intermediate text dump . . . . . : *NONE
Snap dump . . . . . : *NONE
Codelist . . . . . : *NONE
Ignore decimal data error . . . . . : *NO
Actual Program Source:
  Member . . . . . : RSDBAT
  File . . . . . : QICFPUB
  Library . . . . . : ICFLIB
  Last Change . . . . . : 03/20/89 15:26:21
  Description . . . . . : rpg batch file transfer using dds source
5738R61 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDBAT          10/03/90 14:54:22          Page 2
SEQUENCE          IND          DO          PAGE          PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE          NUM          UPDATE          LINE          ID
          Source Listing
100 H*****
200 H*
300 H* THIS IS A BATCH FILE TRANSFER PROGRAM THAT READS A SEQUENTIAL
400 H* FILE AND SENDS THE RECORDS TO THE REMOTE SYSTEM UNTIL THE END
500 H* OF FILE IS REACHED. AT THIS TIME, THE PROGRAM STOPS SENDING
600 H* AND STARTS RECEIVING RECORDS FROM THE REMOTE SYSTEM UNTIL A
700 H* DETACH INDICATION IS RECEIVED.
800 H*
900 H*****
1000 * 1
      H
1100 FSRCICF CF E          WORKSTN          10/14/87
1200 F          KINFDS FEEDBK          10/14/87
      RECORD FORMAT(S): LIBRARY ICFLIB FILE SRCICF.
      EXTERNAL FORMAT RCYDATA RPG NAME RCYDATA
      EXTERNAL FORMAT SNDDATA RPG NAME SNDDATA
      EXTERNAL FORMAT EVOKPGM RPG NAME EVOKPGM
      EXTERNAL FORMAT ENDREC RPG NAME ENDREC
      EXTERNAL FORMAT INVITE RPG NAME INVITE

```

Figure 11-9 (Part 1 of 5). Source Program Example — RSDBAT (User-Defined Formats)


```

1300 FDBFILE IF F 80 DISK 10/14/87
1400 FQPRINT O F 132 PRINTER 10/14/87
1500 IDBFILE NS 80 10/14/87
1600 I 1 80 DBDATA 10/14/87
1700 I* 2 10/14/87
A000000 INPUT FIELDS FOR RECORD RCVDATA FILE SRCICF FORMAT RCVDATA.
A000001 1 80 RCVFLD
B000000 INPUT FIELDS FOR RECORD SNDDATA FILE SRCICF FORMAT SNDDATA.
B000001 1 80 SNDFLD
C000000 INPUT FIELDS FOR RECORD EVOKPGM FILE SRCICF FORMAT EVOKPGM.
D000000 INPUT FIELDS FOR RECORD ENDREC FILE SRCICF FORMAT ENDREC.
E000000 INPUT FIELDS FOR RECORD INVITE FILE SRCICF FORMAT INVITE.
1800 IFEEDBK DS 10/14/87
1900 I 38 45 FMTNM 10/14/87
2000 I 273 282 PGMDEV 10/14/87
2100 I 401 404 MAJMIN 10/14/87
2200 I 401 402 MAJCOD 10/14/87
2300 I 403 404 MINCOD 10/14/87
2400 C*****
2500 C* 03/20/89
2600 C* EVOKE PROGRAM 'RTDBATCL' ON REMOTE SYSTEM IN LIBRARY ICFLIB. 10/14/87
2700 C* INDICATOR 50 (*IN50) IS ASSOCIATED WITH THE EVOKE KEYWORD. 10/14/87
2800 C* 03/20/89
2900 C*****
3000 C* 10/14/87
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSDBAT 10/03/90 14:54:22 Page 3
SEQUENCE NUMBER *...1....+...2....+...3....+...4....+...5....+...6....+...7...* USE IND DO LAST PAGE PROGRAM
UPDATE LINE ID
3100 C ITMIN TAG 10/14/87
3200 C MOVE 'RTDBATCL' PG MID 10/14/87
3300 C MOVE 'ICFLIB ' LIB 10/14/87
3400 C MOVE '1' *IN50 10/14/87
3500 C WRITEEVOKPGM ISSUE EVOKE 10/14/87
3600 C MOVE '0' *IN50 10/14/87
3700 C MAJCOD CABGT'03' NOTOKR ERROR? 10/14/87
3800 C*****
3900 C* 03/20/89
4000 C* AFTER SUCCESSFUL EXECUTION OF THE EVOKE OPERATION, A RECORD 03/20/89
4100 C* IS READ FROM THE DATABASE FILE AND SENT TO THE REMOTE SYSTEM. 03/20/89
4200 C* THIS IS REPEATED UNTIL END OF FILE IS REACHED ON THE DATABASE 03/20/89
4300 C* FILE. AT END OF FILE, THE PROGRAM DEVICE IS INVITED AND 03/20/89
4400 C* CONTROL GOES TO RECDTA TO GET DATA FROM THE REMOTE SYSTEM. 10/16/87
4500 C* 03/20/89
4600 C*****
4700 C* 4 10/14/87
4800 C SENDTA TAG 10/14/87
4900 C READ DBFILE 98 3 10/14/87
5000 C 98 EXSR INVSND INVITE 10/14/87
5100 C EOFPSW IFNE '1' B001 10/14/87
5200 C MOVE DBDATA SNDFLD 001 10/14/87
5300 C EXSR WCFRTN 001 10/14/87
5400 C GOTO SENDTA SEND DATA 001 10/14/87
5500 C END E001 10/14/87
5600 C*****
5700 C* 03/20/89
5800 C* RECEIVE RECORDS FROM THE REMOTE SYSTEM UNTIL THE RCVDETACH 03/20/89
5900 C* INDICATOR IS SET ON. EACH RECORD RECEIVED IS PRINTED TO 03/20/89
6000 C* THE PRINT FILE. 03/20/89
6100 C* 03/20/89
6200 C*****
6300 C* 5 10/14/87
6400 C RECDTA TAG 10/14/87
6500 C READ SRCICF 98 3 10/14/87
6600 C*****
6700 C* 03/20/89
6800 C* IF ANY ICF FILE ERROR OCCURS, PRINT A LINE CONTAINING 03/20/89
6900 C* INFORMATION ABOUT THE ERROR. 10/14/87
7000 C* 03/20/89
7100 C*****
7200 C* 6 10/14/87
7300 C MAJCOD CABGT'03' NOTOKR 10/14/87
7400 C MAJCOD CABEQ'03' CHKDET NO DATA? 10/14/87
7500 C EXCPTPTREC 10/14/87
7600 C CHKDET TAG 10/14/87
7700 C *IN35 CABNE'1' RECDTA DETACH? 10/14/87
7800 C*****
7900 C* 03/20/89
8000 C* AFTER A DETACH INDICATION IS RECEIVED, AN EOJ MESSAGE 03/20/89
8100 C* IS PRINTED AND THE SESSION IS ENDED. 03/20/89
8200 C* 03/20/89

```

Figure 11-9 (Part 2 of 5). Source Program Example — RSDBAT (User-Defined Formats)

```

8300 C*****
8400 C* 7
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSDBAT 10/03/90 14:54:22 Page 4
SEQUENCE LAST PAGE PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE NUM UPDATE LINE ID
8500 C EXCPTOKEND 10/14/87
8600 C GOT0 END 10/14/87
8700 C*****
8800 C*
8900 C* WHEN AN ERROR OCCURS ON AN ICF SESSION, INFORMATION 03/20/89
9000 C* ABOUT THE ERROR IS PRINTED. 03/20/89
9100 C*
9200 C*****
9300 C* 8
9400 C NOTOKR TAG 10/14/87
9500 C EXCPTNOTOK 10/14/87
9600 C*****
9700 C*
9800 C* WHEN PROCESSING IS FINISHED, THE LAST RECORD SWITCH 10/14/87
9900 C* IS TURNED ON AND THE PROGRAM IS ENDED. 10/14/87
10000 C*
10100 C*****
10200 C* 9
10300 C END TAG 10/14/87
10400 C SETON LR 1 10/14/87
10500 C RETRN 10/14/87
10600 C*****
10700 C*
10800 C* THIS SUBROUTINE SENDS DATA TO THE REMOTE SYSTEM. 10/14/87
10900 C*
11000 C*****
11100 C* 10
11200 C WCFRTN BEGSR 10/14/87
11300 C WRITESNDDATA 10/14/87
11400 C MAJCOD CABGT'03' NOTOKR 10/14/87
11500 C ENDSR 10/14/87
11600 C*****
11700 C*
11800 C* THIS SUBROUTINE IS CALLED AT END OF FILE TO REQUEST THE REMOTE 10/14/87
11900 C* PROGRAM TO START SENDING DATA. AN INVITE OPERATION IS ISSUED 10/14/87
12000 C* TO NOTIFY THE TARGET PROGRAM THAT IT CAN START SENDING DATA. 10/14/87
12100 C*
12200 C*****
12300 C* 11
12400 C INVSND BEGSR 10/14/87
12500 C MOVE '1' EOFPSW 1 10/14/87
12600 C MOVE '1' *IN45 10/14/87
12700 C WRITEINVITE 10/14/87
12800 C MAJCOD CABGT'03' NOTOKR 10/14/87
12900 C ENDSR 10/14/87
13000 C*****
13100 OOPRINT E 1 PTREC 10/14/87
13200 0 RCVFLD 80 10/14/87
13300 0 E 1 OKEND 10/14/87
13400 0 21 'RSDBAT HAS COMPLETED ' 10/14/87
13500 0 30 'NORMALLY.' 10/14/87
13600 0 E 1 NOTOK 10/14/87
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSDBAT 10/03/90 14:54:22 Page 5
SEQUENCE LAST PAGE PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE NUM UPDATE LINE ID
13700 0 21 'RSDBAT HAS COMPLETED ' 10/14/87
13800 0 32 'ABNORMALLY.' 10/14/87
13900 0 MAJCOD 35 10/14/87
14000 0 36 '/' 10/14/87
14100 0 MINCOD 39 10/14/87
14200 0 46 'FORMAT:' 10/14/87
14300 0 FMTNM 56 10/14/87
14400 0 63 'DEVICE:' 10/14/87
14500 0 PGMDEV 80 03/20/89
* 6103 14501 OVERFLOW INDICATOR 0A ASSIGNED TO FILE OPRINT.
F000000 OUTPUT FIELDS FOR RECORD SNDDATA FILE SRCICF FORMAT SNDDATA.
F000001 SNDFLD 80 CHAR 80
G000000 OUTPUT FIELDS FOR RECORD EVOKPGM FILE SRCICF FORMAT EVOKPGM.
G000001 PGMID 10 CHAR 10
G000002 LIB 20 CHAR 10
H000000 OUTPUT FIELDS FOR RECORD INVITE FILE SRCICF FORMAT INVITE.
***** E N D O F S O U R C E *****

```

Figure 11-9 (Part 3 of 5). Source Program Example — RSDBAT (User-Defined Formats)

```

Additional Diagnostic Messages
* 7089      1100  RPG PROVIDES SEPARATE INDICATOR AREA FOR FILE SRCICF.
* 7086      1300  RPG PROVIDES BLOCK OR UNBLOCK SUPPORT FOR FILE DBFILE.
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDBAT          10/03/90 14:54:22          Page          6
                C r o s s   R e f e r e n c e
File and Record References:
FILE/RCD      DEV/RCD      REFERENCES (D=DEFINED)
 02 DBFILE     DISK          1300D   1500   4900
 03 QPRINT     PRINTER         1400D  13100  13300   13600  14501
 01 SRCICF     WORKSTN         1100D   6500
      ENDREC         1100D D000000
      EVOKPGM        1100D C000000   3500  G000000
      INVITE         1100D E000000  12700  H000000
      RCVDATA        1100D A000000
      SNDDATA        1100D B000000  11300  F000000
Field References:
FIELD         ATTR      REFERENCES (M=MODIFIED D=DEFINED)
*IN35        A(1)       7700
*IN45        A(1)      12600M
*IN50        A(1)      3400M   3600M
CHKDET       TAG        7400    7600D
DBDATA       A(80)     1600D   5200
END          TAG        8600    10300D
EOFPSW       A(1)      5100    12500D
FEEDBK       DS(404)    1100    1800D
FMTNM        A(8)      1900D   14300
INVSND       BEGSR     5000    12400D
* 7031 ITMIN       TAG        3100D
LIB          A(10)     3300M  G000002D
MAJCOD       A(2)      2200D   3700   7300   7400   11400
                12800   13900
* 7031 MAJMIN       A(4)      21600D
MINCOD       A(2)      2300D   14100
NOTOK        EXCPT     9500    13600D
NOTOKR       TAG        3700    7300   9400D  11400  12800
OKEND        EXCPT     8500    13300D
PGMDEV       A(10)     2000D   14500
PGMID        A(10)     3200M  G000001D
PTREC        EXCPT     7500    13100D
RCVFLD       A(80)     A000001D 13200
RECDTA       TAG        6400D   7700
SENDTA       TAG        4800D   5400
SNDFLD       A(80)     B000001D 5200M F000001D
WCFRTN       BEGSR     5300    11200D
'ICFLIB '    LITERAL   3300
'RTDBATCL'   LITERAL   3200
'0'          LITERAL   3600
'03'         LITERAL   3700    7300   7400   11400  12800
'1'          LITERAL   3400    5100   7700   12500  12600
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDBAT          10/03/90 14:54:22          Page          7
Indicator References:
INDICATOR     REFERENCES (M=MODIFIED D=DEFINED)
*1N           3400M   3600M   7700   12600M
LR            10400M
OA            1400D   14501
* 7031 15
* 7031 30
          35          7700
          45         12600M
          50         3400M   3600M
* 7031 80         1500M
          98         4900M   5000   6500M
* * * * * E N D   O F   C R O S S   R E F E R E N C E   * * * * *
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDBAT          10/03/90 14:54:22          Page          8
                M e s s a g e   S u m m a r y
* QR66103 Severity: 00 Number: 1
  Message . . . : No Overflow Indicator is specified but an
  indicator is assigned to a file and automatic skip to 6 is
  generated.
* QR67031 Severity: 00 Number: 5
  Message . . . : The Name or indicator is not referenced.
* QR67086 Severity: 00 Number: 1
  Message . . . : The RPG handles blocking function for file.
  INFDS contents updated only when blocks of data transferred.
* QR67089 Severity: 00 Number: 1
  Message . . . : The RPG provides Separate-Indicator area for
  file.
* * * * * E N D   O F   M E S S A G E   S U M M A R Y   * * * * *
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDBAT          10/03/90 14:54:22          Page          9

```

Figure 11-9 (Part 4 of 5). Source Program Example — RSDBAT (User-Defined Formats)

```

                          F i n a l   S u m m a r y
Message Count: (by Severity Number)
      TOTAL   00   10   20   30   40   50
           8    8    0    0    0    0    0
Program Source Totals:
Records . . . . . : 145
Specifications . . . . . : 66
Table Records . . . . . : 0
Comments . . . . . : 79
PRM has been called.
Program RSDBAT is placed in library ICFLIB. 00 highest Error-Severity-Code.
***** END OF COMPILATION *****

```

Figure 11-9 (Part 5 of 5). Source Program Example — RSDBAT (User-Defined Formats)

```

5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSFBAT          10/03/90 14:55:47      Page      1
Compiler . . . . . : IBM AS/400 RPG/400
Command Options:
  Program . . . . . : ICFLIB/RSFBAT
  Source file . . . . . : ICFLIB/QICFPUB
  Source member . . . . . : *PGM
Text not available for message RXT0073 file QRPMSG.
  Generation options . . . . . : *NOLIST *NOXREF *NOATR *NODUMP *NOOPTIMIZE
  Source listing indentation . . . . . : *NONE
  SAA flagging . . . . . : *NOFLAG
  Generation severity level . . . . . : 9
  Print file . . . . . : *LIBL/QSPRT
  Replace program . . . . . : *YES
  Target release . . . . . : *CURRENT
  User profile . . . . . : *USER
  Authority . . . . . : *LIBCRTAUT
  Text . . . . . : *SRCMBRTXT
  Phase trace . . . . . : *NO
  Intermediate text dump . . . . . : *NONE
  Snap dump . . . . . : *NONE
  Codelist . . . . . : *NONE
  Ignore decimal data error . . . . . : *NO

```

Actual Program Source:

```

Member . . . . . : RSFBAT
File . . . . . : QICFPUB
Library . . . . . : ICFLIB
Last Change . . . . . : 03/20/89 15:30:19
Description . . . . . : rpg batch file transfer using $$FORMAT
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSFBAT          10/03/90 14:55:47      Page      2

```

SEQUENCE NUMBER	INDDO	DO LAST	DO LAST	PAGE	PROGRAM
...1...+...2...+...3...+...4...+...5...+...6...+...7...	USE	NUM	UPDATE	LINE	ID
Source Listing					
100	H*				10/16/87
200	H*				03/20/89
300	H*				10/16/87
400	H*				10/16/87
500	H*				10/16/87
600	H*				10/16/87
700	H*				10/16/87
800	H*				03/20/89
900	H*				10/16/87
1000	H* 1				10/16/87
1100	FSRCICF	CF	F	84	WORKSTN
1200	F				KINFDS FEEDBK
1300	FDBFILE	IF	F	80	DISK
1400	FQPRINT	0	F	132	PRINTER
1500	ISRCICF	NS	82		
1600	I				1 80 RCVFLD
1700	IDBFILE	NS	80		
1800	I				1 80 DBDATA
1900	I* 2				
2000	IFEEDBK				DS
2100	I				38 45 FMTNM
2200	I				273 282 PGMDEV
2300	I				401 404 MAJMIN
2400	I				401 402 MAJCOD
2500	I				403 404 MINCOD
2600	C*				
2700	C*				03/20/89
2800	C*				EVOKE PROGRAM 'RTFBATCL' ON REMOTE SYSTEM IN LIBRARY ICFLIB.
2900	C*				THE USER ID AND PASSWORD ARE DEFINED AS PART OF THE \$\$EVOKNI
3000	C*				FORMAT.
3100	C*				
3200	C*				
3300	C* 3				
3400	C				ITMIN TAG
3500	C				EXCPTEVOKE
3600	C				MAJCOD CABGT'03' NOTOKR
3700	C*				ISSUE EVOKE ERROR?
3800	C*				
3900	C*				03/20/89
4000	C*				AFTER THE SUCCESSFUL EXECUTION OF THE EVOKE OPERATION, A
4100	C*				RECORD IS READ FROM THE DATABASE FILE, AND SENT TO THE REMOTE
4200	C*				SYSTEM. THIS IS REPEATED UNTIL AN END OF FILE IS REACHED ON
4300	C*				THE DATABASE FILE. AT THIS TIME, THE PROGRAM DEVICE IS INVI-
4400	C*				TED, AND CONTROL GOES TO RECDTA TO GET DATA FROM THE REMOTE
4500	C*				SYSTEM.

Figure 11-10 (Part 1 of 4). Source Program Example — RSFBAT (System-Supplied Formats)

```

4600 C*****
4700 C* 4 10/16/87
4800 C SENDTA TAG 10/16/87
4900 C READ DBFILE 98 3 10/16/87
5000 C 98 EXSR INVSND INVITE 10/16/87
5100 C EOFPSW IFNE '1' B001 10/16/87
5200 C EXSR WCFRTN 001 10/16/87
5300 C GOTO SENDTA SEND DATA 001 10/16/87
5400 C END E001 10/16/87
5500 C*****
5600 C* 03/20/89
5700 C* THE PROGRAM STARTS RECEIVING RECORDS AT THIS POINT FROM THE 03/20/89
5800 C* REMOTE SYSTEM UNTIL A DETACH INDICATION IS RECEIVED. EACH 03/20/89
5900 C* RECORD RECEIVED IS PRINTED TO THE PRINT FILE. 03/20/89
6000 C* 03/20/89
6100 C*****
6200 C* 5 10/16/87
6300 C RECDTA TAG 10/16/87
6400 C READ SRCICF 98 3 10/16/87
6500 C*****
6600 C* 03/20/89
6700 C* IF AN ICF FILE ERROR OCCURS, PRINT A LINE CONTAINING 03/20/89
6800 C* INFORMATION ABOUT THE ERROR. 10/16/87
6900 C* 03/20/89
7000 C*****
7100 C* 6 10/16/87
7200 C MAJCOD CABGT'03' NOTOKR 10/16/87
7300 C MAJCOD CABEQ'03' CHKDET NO DATA? 10/16/87
7400 C EXCPTPTREC 10/16/87
7500 C CHKDET TAG 10/16/87
7600 C MINCOD CABNE'08' RECDTA DETACH? 10/16/87
7700 C*****
7800 C* 03/20/89
7900 C* AFTER A DETACH INDICATION IS RECEIVED, AN EOJ MESSAGE IS 03/20/89
8000 C* PRINTED AND THE SESSION IS ENDED. 03/20/89
8100 C* 03/20/89
8200 C*****
8300 C* 7 10/16/87
8400 C EXCPTOKEND 10/16/87
8500 C GOTO END 10/16/87
8600 C*****
8700 C* 03/20/89
8800 C* WHEN AN ERROR OCCURS ON AN ICF SESSION, INFORMATION 03/20/89
8900 C* ABOUT THE ERROR IS PRINTED. 03/20/89
9000 C* 03/20/89
9100 C*****
9200 C* 8 10/16/87
9300 C NOTOKR TAG 10/16/87
9400 C EXCPTNOTOK 10/16/87
9500 C*****
9600 C* 03/20/89
9700 C* WHEN PROCESSING IS FINISHED, THE LAST RECORD SWITCH IS 03/20/89
9800 C* TURNED ON AND THE PROGRAM IS ENDED. 03/20/89
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICLIB/RSFBAT 10/03/90 14:55:47 Page 4
SEQUENCE IND DO LAST PAGE PROGRAM
NUMBER *...1....+...2....+...3....+...4....+...5....+...6....+...7...* USE NUM UPDATE LINE ID
9900 C* 03/20/89
10000 C*****
10100 C* 9 10/16/87
10200 C END TAG 10/16/87
10300 C SETON LR 1 10/16/87
10400 C RETRN 10/16/87
10500 C*****
10600 C* 03/20/89
10700 C* THIS SUBROUTINE SENDS DATA TO THE REMOTE SYSTEM. 10/16/87
10800 C* 03/20/89
10900 C*****
11000 C* 10 10/16/87
11100 C WCFRTN BEGSR 10/16/87
11200 C EXCPTSNDATA 10/16/87
11300 C MAJCOD CABGT'03' NOTOKR 10/16/87
11400 C ENDSR 10/16/87
11500 C*****
11600 C* 03/20/89
11700 C* THIS SUBROUTINE IS CALLED AT END OF FILE TO REQUEST THE REMOTE 10/16/87
11800 C* PROGRAM TO START SENDING DATA. AN INVITE OPERATION IS ISSUED 10/16/87
11900 C* TO NOTIFY THE TARGET PROGRAM THAT IT CAN START SENDING DATA. 10/16/87
12000 C* 03/20/89
12100 C*****

```

Figure 11-10 (Part 2 of 4). Source Program Example — RSFBAT (System-Supplied Formats)

```

12200 C* 11
12300 C          INVSND  BEGSR
12400 C          MOVE '1'      EOFPSW 1
12500 C          EXCPTINVITE
12600 C          MAJCOD  CABGT'03'  NOTOKR
12700 C          ENDSR
12800 C*****
12900 OQPRINT  E 1          PTREC
13000 0          RCVFLD  80
13100 0          E 1          OKEND
13200 0          21 'RSFBAT HAS COMPLETED '
13300 0          30 'NORMALLY.'
13400 0          E 1          NOTOK
13500 0          21 'RSFBAT HAS COMPLETED '
13600 0          32 'ABNORMALLY.'
13700 0          MAJCOD  35
13800 0          36 '/'
13900 0          MINCOD  39
14000 0          46 'FORMAT:'
14100 0          FMTNM   56
14200 0          63 'DEVICE:'
14300 0          PGMDEV  80
14400 OSRCICF  E          EVOKE
14500 0          K8 '$$EVOKNI'
14600 0          8 'RTFBATCL'
14700 0          16 'QSECOFR '
14800 0          24 'QSECOFR '
14900 0          32 'ICFLIB '
15000 0          E          SNDATA
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSFBAT 10/03/90 14:55:47 Page 5
SEQUENCE NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE IND DO LAST PAGE PROGRAM
NUMBER          *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE IND DO LAST PAGE PROGRAM
15100 0          K8 '$$SENDNI'
15200 0          4 '0000'
15300 0          DBDATA  84
15400 0          E          INVITE
15500 0          K6 '$$SEND'
15600 0          4 '0000'
* 6103 15601 OVERFLOW INDICATOR 0A ASSIGNED TO FILE QPRINT.
      ***** END OF SOURCE *****
      Additional Diagnostic Messages
* 7089 1100 RPG PROVIDES SEPARATE INDICATOR AREA FOR FILE SRCICF.
* 7086 1300 RPG PROVIDES BLOCK OR UNBLOCK SUPPORT FOR FILE DBFILE.
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSFBAT 10/03/90 14:55:47 Page 6
      Cross Reference
File and Record References:
FILE/RCD DEV/RCD REFERENCES (D=DEFINED)
02 DBFILE DISK 13000 1700 4900
03 QPRINT PRINTER 14000 12900 13100 13400 15601
01 SRCICF WORKSTN 11000 1500 6400 14400 15000 15400
  $$EVOKNI 14500
  $$SEND 15500
  $$SENDNI 15100
Field References:
FIELD ATTR REFERENCES (M=MODIFIED D=DEFINED)
CHKDET TAG 7300 7500D
DBDATA A(80) 18000 15300
END TAG 8500 10200D
EOFPSW A(1) 5100 12400D
EVOKE EXCPT 3500 14400
FEEDBK DS(404) 1100 2000D
FMTNM A(8) 21000 14100
INVITE EXCPT 12500 15400
INVSND BEGSR 5000 12300D
* 7031 ITMIN TAG 34000
MAJCOD A(2) 24000 3600 7200 7300 11300 12600 13700
* 7031 MAJMIN A(4) 23000
MINCOD A(2) 25000 7600 13900
NOTOK EXCPT 9400 13400
NOTOKR TAG 3600 7200 9300D 11300 12600
OKEND EXCPT 8400 13100
PGMDEV A(10) 22000 14300
PTREC EXCPT 7400 12900
RCVFLD A(80) 16000 13000
RECDTA TAG 63000 7600
SENDA TAG 48000 5300
SNDATA EXCPT 11200 15000
WCFRTN BEGSR 5200 11100D
'03' LITERAL 3600 7200 7300 11300 12600
'08' LITERAL 7600

```

Figure 11-10 (Part 3 of 4). Source Program Example — RSFBAT (System-Supplied Formats)

```

'1'          LITERAL 5100 12400
Indicator References:
INDICATOR REFERENCES (M=MODIFIED D=DEFINED)
LR          10300M
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSFBAT          10/03/90 14:55:47          Page 7
OA          1400D 15601
* 7031 80          1700M
* 7031 82          1500M
98          4900M 5000 6400M
***** END OF CROSS REFERENCE *****
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSFBAT          10/03/90 14:55:47          Page 8
Message Summary
* QR66103 Severity: 00 Number: 1
Message . . . . : No Overflow Indicator is specified but an
indicator is assigned to a file and automatic skip to 6 is
generated.
* QR67031 Severity: 00 Number: 4
Message . . . . : The Name or indicator is not referenced.
* QR67086 Severity: 00 Number: 1
Message . . . . : The RPG handles blocking function for file.
INFDS contents updated only when blocks of data transferred.
* QR67089 Severity: 00 Number: 1
Message . . . . : The RPG provides Separate-Indicator area for
file.
***** END OF MESSAGE SUMMARY *****
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSFBAT          10/03/90 14:55:47          Page 9
Final Summary
Message Count: (by Severity Number)
TOTAL 00 10 20 30 40 50
7 7 0 0 0 0
Program Source Totals:
Records . . . . . : 156
Specifications . . . . . : 75
Table Records . . . . . : 0
Comments . . . . . : 81
PRM has been called.
Program RSFBAT is placed in library ICFLIB. 00 highest Error-Severity-Code.
***** END OF COMPILATION *****

```

Figure 11-10 (Part 4 of 4). Source Program Example — RSFBAT (System-Supplied Formats)

Target Program Batch Transfer

(Example 1): The following describes an RPG/400 batch data transfer target program.

Program Files: The RPG/400 batch transfer target program uses the following files:

TGTICF

An ICF file used to send records to and receive records from the source program.

DBFILE

A database file that contains the records to be sent to the source program.

QPRINT

A printer file used to print the records received from the source program.

DDS Source: The DDS used in the ICF file is illustrated in the following example. The other files (DBFILE and QPRINT) are program-described and therefore do not require DDS.

```

A*****
A*                                     *
A*          ICF FILE                   *
A*      USED IN BATCH DATA TRANSFER PROGRAM *
A*                                     *
A*****
A* FILE LEVEL INDICATORS:
A*
A*                                     INDARA
A*
A*          RCVTRNRND(15 'END OF DATA')
A*
A* 30          DETACH
A*
A*          INDTXT(30 '30->DETACH TARG-
A*          ET PROGRAM.')
```

```

A*          RCVDDETACH(35 'RECEIVED -
A*          DETACHED.')
```

```

A*****
A*          ICF RECORD FORMATS          *
A*****
R RCVDATA
R CVFLD          88A
R SNDDATA
R SNDFLD          88A
R EVOKPGM
A 50          EVOKE(&LIB/&PGMID)
A 50          SECURITY(2 'PASSWRD' +
                3 'USERID')
```

```

A          PGMID          10A P
A          LIB            10A P
A          R ENDREC
A          R INVITE
A 45          INVITE
```

This example acquires all program devices at the beginning of the program. For performance considerations, you may not want to acquire program devices until they are actually needed in the program.

ICF File Creation and Program Device Entry

Definition: The command needed to create the ICF file is:

```

CRTICFF FILE(ICFLIB/TGTICF) SRCFILE(ICFLIB/QICFPUB)
SRCMBR(TGTICF) ACQPGMDEV(PGMDEVB)
TEXT('TARGET ICF FILE FOR BATCH DATA TRANSFER')
```

The command needed to define the program device entry is:

```

ADDICFDEVE FILE(ICFLIB/TGTICF) PGMDEV(PGMDEVB)
RMTLOCNAME(*REQUESTER)
```

Program Explanation: The following describes the structure of the program examples illustrated in Figure 11-11 on page 11-19 and Figure 11-12 on page 11-23. The ICF file used in the first example is defined by the user and uses externally described data formats (DDS). The second example uses the same file, but uses program-described data and system-supplied formats. The reference letters in the explanation below correspond to those in the following program examples.

Although the basic structure of the two examples provided is the same, there are differences because of the use of user-defined formats and system-supplied formats. All output operations to the ICF file in the first example are done using the WRITE statement. All output operations to the ICF file in the second example using system-supplied formats are done using the EXCPT statement.

Differences between the first and second example are described in notes in each of the descriptions.

- 1 The file specification identifies the files used in the program. TGTICF is the ICF file used to send records to the source program.

The files used in the program are opened at the beginning of the RPG/400 cycle and the ICF program device is implicitly acquired because the ACQPGMDEV parameter was specified on the CRTICFF command.

Note: In the program using system-supplied formats, the input records for TGTICF are explicitly coded since TGTICF is treated as a program-described file. The system-supplied file, QICDMF, can be used instead of TGTICF. Using the system-supplied file is done by specifying QICDMF in the file specification, or by using an OVRICFF command to change the file name from TGTICF to QICDMF. The OVRICFF

command can also be used to change the ACQPGMDEV parameter of the file.

- 2** FEEDBK is the name of the file information data structure (INFDS) used with TGTICF. It contains the following information:

- Record format-name (FMTNM)
- Program device name (PGMDEV)
- Major/minor return code (MAJMIN, MAJCOD, MINCOD)

- 3** Read data from the ICF program device (TGTICF) file.

If an error occurs on the read (major return code greater than 03), control passes to **6**. Otherwise, if data is received (major return code not = 03), the data is written to the printer file (QPRINT).

Data records are read until the change-direction indication is received from the source program. When change direction is received, indicator 15 is set on, as defined by the RCVTRNRND keyword in the DDS for the ICF file, and control is passed to **4**.

Note: In the program using system-supplied formats, the minor return code of '00' is checked to verify whether change direction is received.

- 4** The database file is read and the records sent to the source program until the end of the database file. At this time, the program sets indicator 98 and goes to **9**. After returning from **9**, control is passed to **5**.

If it is not the last record, the data is moved to field SNDFLD, and the program goes to **8** to write the record to the ICF program device. When control returns from **8**, the next database record is read.

- 5** After the last database record has been read, the following message is written to the print file:

```
RTDBAT HAS COMPLETED NORMALLY
```

Control passes to **7**.

Note: The program name is RSFBAT in the program using system-supplied formats.

- 6** When an I/O operation to the ICF file (TGTICF) completes unsuccessfully, the following message is written to the print file:

```
RTDBAT HAS COMPLETED ABNORMALLY
```

Control passes to **7**.

Note: The program name is RTFBAT in the program using system-supplied formats.

- 7** The program ends the job by setting on LR indicator and returning to caller of the program. The ICF file is closed and the session is ended at the end of the RPG cycle.

- 8** This subroutine is called to write data to the ICF program device using the format SNDDATA. If an error occurs, the program goes to **6** and a message is printed.

Note: The \$\$\$SENDNI format is used instead of the user-defined SNDDATA format in the program using system-supplied formats.

- 9** This subroutine is called to issue a detach request to the ICF program device using format ENDREC. If an error occurs, the program goes to **6** and a message is printed.

Note: The \$\$\$SENDET format is used instead of the user-defined ENDREC format in the program using system-supplied formats.

```

5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTDBAT          10/03/90 14:54:38      Page 1
Compiler . . . . . : IBM AS/400 RPG/400
Command Options:
  Program . . . . . : ICFLIB/RTDBAT
  Source file . . . . . : ICFLIB/QICFPUB
  Source member . . . . . : *PGM
Text not available for message RXT0073 file QRPMSG.
  Generation options . . . . . : *NOLIST *NOXREF *NOATR *NODUMP *NOOPTIMIZE
  Source listing indentation . . . . . : *NONE
  SAA flagging . . . . . : *NOFLAG
  Generation severity level . . . . . : 9
  Print file . . . . . : *LIBL/QSYSRPT
  Replace program . . . . . : *YES
  Target release . . . . . : *CURRENT
  User profile . . . . . : *USER
  Authority . . . . . : *LIBCRTAUT
  Text . . . . . : *SRCBRTXT
  Phase trace . . . . . : *NO
  Intermediate text dump . . . . . : *NONE
  Snap dump . . . . . : *NONE
  Codelist . . . . . : *NONE
  Ignore decimal data error . . . . . : *NO

```

```

Actual Program Source:
Member . . . . . : RTDBAT
File . . . . . : QICFPUB
Library . . . . . : ICFLIB
Last Change . . . . . : 03/20/89 15:40:57
Description . . . . . : rpg batch file transfer using dds source
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTDBAT          10/03/90 14:54:38      Page 2
SEQUENCE

```

NUMBER	SEQUENCE	IND	DO	LAST	PAGE	PROGRAM
		NUM	NUM	UPDATE	LINE	ID
100	H*****			10/16/87		
200	H*			03/20/89		
300	H* THIS PROGRAM IS EVOKED BY THE SOURCE PROGRAM AND RECEIVES			03/20/89		
400	H* RECORDS FROM IT. WHEN THE SOURCE PROGRAM IS DONE SENDING			03/20/89		
500	H* DATA, THIS PROGRAM SENDS ITS OWN RECORDS. WHEN FINISHED,			03/20/89		
600	H* THIS PROGRAM WILL SEND A DETACH REQUEST TO THE SOURCE			03/20/89		
700	H* PROGRAM TO END THE SESSION AND JOB.			03/20/89		
800	H*			03/20/89		
900	H*****			10/16/87		
1000	* 1			10/16/87		
	H					*****
1100	FTGTICF CF E			10/16/87		
1200	F			10/16/87		
	RECORD FORMAT(S): LIBRARY ICFLIB FILE TGTICF.					
	EXTERNAL FORMAT RCVDATA RPG NAME RCVDATA					
	EXTERNAL FORMAT SNDDATA RPG NAME SNDDATA					
	EXTERNAL FORMAT EVOKPGM RPG NAME EVOKPGM					
	EXTERNAL FORMAT ENDREC RPG NAME ENDREC					
	EXTERNAL FORMAT INVITE RPG NAME INVITE					
1300	FDBFILE IF F 80			10/16/87		
1400	FQPRINT O F 132			10/16/87		
1500	IDBFILE NS 80			10/16/87		
1600	I			10/16/87		
1700	I* 2			10/16/87		
A000000	INPUT FIELDS FOR RECORD RCVDATA FILE TGTICF FORMAT RCVDATA.					
A000001	1 80 RCVFLD					
B000000	INPUT FIELDS FOR RECORD SNDDATA FILE TGTICF FORMAT SNDDATA.					
B000001	1 80 SNDFLD					
C000000	INPUT FIELDS FOR RECORD EVOKPGM FILE TGTICF FORMAT EVOKPGM.					
D000000	INPUT FIELDS FOR RECORD ENDREC FILE TGTICF FORMAT ENDREC.					
E000000	INPUT FIELDS FOR RECORD INVITE FILE TGTICF FORMAT INVITE.					
1800	IFEEBKB DS			10/16/87		
1900	I			10/16/87		
2000	I			10/16/87		
2100	I			10/16/87		
2200	I			10/16/87		
2300	I			10/16/87		
2400	C*****			10/16/87		
2500	C*			03/20/89		
2600	C* THIS PROGRAM ISSUES A READ OPERATION TO THE PROGRAM DEVICE			03/20/89		
2700	C* TO RECEIVE RECORDS FROM THE SOURCE PROGRAM UNTIL THE			03/20/89		
2800	C* RCYTRNRD INDICATOR (*IN15) IS SET. EACH RECORD RECEIVED IS			03/20/89		
2900	C* PRINTED TO THE PRINT FILE.			03/20/89		
3000	C*			10/16/87		

```

5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTDBAT          10/03/90 14:54:38      Page 3
SEQUENCE
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE
3100 C* IF AN ERROR OCCURS, AN ERROR MESSAGE IS PRINTED AND THE

```

Figure 11-11 (Part 1 of 4). Target Program Example — RTDBAT (User-Defined Formats)

```

3200 C* JOB IS ENDED. 10/16/87
3300 C* 03/20/89
3400 C***** 10/16/87
3500 C* 3 10/16/87
3600 C RECDTA TAG 10/16/87
3700 C READ TGTICF 98 3 10/16/87
3800 C MAJCOD CABGT'03' NOTOKR ERROR? 10/16/87
3900 C MAJCOD CABEQ'03' CHKTRN NO DATA ? 10/16/87
4000 C EXCPTPTREC 10/16/87
4100 C CHKTRN TAG 10/16/87
4200 C *IN15 CABNE'1' RECDTA RCVTRNRND ? 10/16/87
4300 C***** 10/16/87
4400 C* 03/20/89
4500 C* WHEN A RCVTRNRND INDICATION IS RECEIVED, THE PROGRAM STARTS 10/16/87
4600 C* SENDING THE RECORDS TO THE SOURCE PROGRAM. RECORDS ARE SENT 10/16/87
4700 C* UNTIL AN END OF FILE IS REACHED ON THE DATABASE FILE. AT 03/20/89
4800 C* THIS TIME, A DETACH REQUEST IS SENT TO THE SOURCE PROGRAM. 03/20/89
4900 C* 03/20/89
5000 C***** 10/16/87
5100 C* 4 10/16/87
5200 C SENDTA TAG 10/16/87
5300 C READ DBFILE 98 3 10/16/87
5400 C 98 EXSR ENDS 10/16/87
5500 C EOFPSW IFNE '1' SEND DETACH 10/16/87
5600 C MOVE DBDATA SNDFLD B001 10/16/87
5700 C EXSR WCFRTN 001 10/16/87
5800 C GOTO SENDTA SEND DATA 001 10/16/87
5900 C END E001 10/16/87
6000 C***** 10/16/87
6100 C* 03/20/89
6200 C* WHEN THE END OF FILE IS REACHED, AN EOJ MESSAGE IS 03/20/89
6300 C* PRINTED AND THE PROGRAM GOES TO END. 03/20/89
6400 C* 03/20/89
6500 C***** 10/16/87
6600 C* 5 10/16/87
6700 C EXCPTOKEND 10/16/87
6800 C GOTO END 10/16/87
6900 C***** 10/16/87
7000 C* 03/20/89
7100 C* WHEN AN I/O OPERATION ERROR IS DETECTED, AN ABNORMAL 10/16/87
7200 C* TERMINATION MESSAGE IS PRINTED AND THE PROGRAM ENDS. 10/16/87
7300 C* 03/20/89
7400 C***** 10/16/87
7500 C* 6 10/16/87
7600 C NOTOKR TAG 10/16/87
7700 C EXCPTNOTOK 10/16/87
7800 C***** 10/16/87
7900 C* 03/20/89
8000 C* WHEN PROCESSING IS FINISHED, THE LAST RECORD SWITCH IS SET 10/16/87
8100 C* AND THE PROGRAM IS ENDED. 10/16/87
8200 C* 03/20/89
8300 C***** 10/16/87
8400 C* 7 10/16/87
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICLIB/RTDBAT 10/03/90 14:54:38 Page 4
SEQUENCE LAST PAGE PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE NUM UPDATE LINE ID
8500 C END TAG 10/16/87
8600 C SETON LR 1 10/16/87
8700 C RETRN 10/16/87
8800 C***** 10/16/87
8900 C* 03/20/89
9000 C* THIS SUBROUTINE IS CALLED TO SEND DATA TO THE SOURCE PRO- 03/20/89
9100 C* GRAM. IF A SESSION ERROR OCCURS, AN ABNORMAL TERMINATION 03/20/89
9200 C* MESSAGE IS PRINTED, THE LR SWITCH IS SET, AND THE JOB ENDS. 03/20/89
9300 C* 03/20/89
9400 C***** 10/16/87
9500 C* 8 10/16/87
9600 C WCFRTN BEGSR 10/16/87
9700 C WRITESNDDATA 10/16/87
9800 C MAJCOD CABGT'03' NOTOKR ERROR? 10/16/87
9900 C ENDSR 10/16/87
10000 C***** 10/16/87
10100 C* 03/20/89
10200 C* THIS SUBROUTINE IS CALLED AT END OF FILE TO SEND AN 10/16/87
10300 C* INDICATION TO THE SOURCE SYSTEM THAT TRANSMISSION IS ENDED. 03/20/89
10400 C* THE END OF FILE SWITCH IS ALSO SET TO END THE JOB. 10/16/87
10500 C* 03/20/89
10600 C***** 10/16/87

```

Figure 11-11 (Part 2 of 4). Target Program Example — RTDBAT (User-Defined Formats)

```

10700 C* 9 10/16/87
10800 C ENDSER BEGSR 10/16/87
10900 C MOVE '1' *IN30 ACTV DETACH 10/16/87
11000 C MOVE '1' EOFPSW 1 10/16/87
11100 C WRITEENDREC SEND DETACH 10/16/87
11200 C MAJCOD CABGT'03' NOTOKR ERROR? 10/16/87
11300 C ENDSR 10/16/87
11400 C***** 10/16/87
11500 OQPRINT E 1 PTREC 10/16/87
11600 0 RCVFLD 80 10/16/87
11700 0 E 1 OKEND 10/16/87
11800 0 21 'RTDBAT HAS COMPLETED ' 10/16/87
11900 0 30 'NORMALLY.' 10/16/87
12000 0 E 1 NOTOK 10/16/87
12100 0 21 'RTDBAT HAS COMPLETED ' 10/16/87
12200 0 32 'ABNORMALLY.' 10/16/87
12300 0 MAJCOD 37 10/16/87
12400 0 38 '/' 10/16/87
12500 0 MINCOD 40 10/16/87
12600 0 49 'FORMAT:' 10/16/87
12700 0 FMTNM 60 10/16/87
12800 0 69 'DEVICE:' 10/16/87
12900 0 PGMDEV 80 03/20/89
* 6103 12901 OVERFLOW INDICATOR OA ASSIGNED TO FILE OPRINT.
F000000 OUTPUT FIELDS FOR RECORD SNDDATA FILE TGTCIF FORMAT SNDDATA.
F000001 SNDFLD 80 CHAR 80
6000000 OUTPUT FIELDS FOR RECORD ENDREC FILE TGTCIF FORMAT ENDREC.
***** E N D O F S O U R C E *****
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RTDBAT 10/03/90 14:54:38 Page 5
Additional Diagnostic Messages
* 7089 1100 RPG PROVIDES SEPARATE INDICATOR AREA FOR FILE TGTCIF.
* 7086 1300 RPG PROVIDES BLOCK OR UNBLOCK SUPPORT FOR FILE DBFILE.
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RTDBAT 10/03/90 14:54:38 Page 6
Cross Reference
File and Record References:
FILE/RCD DEV/RCD REFERENCES (D=DEFINED)
02 DBFILE DISK 13000 1500 5300
03 OPRINT PRINTER 14000 11500 11700 12000 12901
01 TGTCIF WORKSTN 11000 3700
ENDREC 11000 D000000 11100 6000000
EVOKPGM 11900 C000000
INVITE 11000 E000000
RCVDATA 11000 A000000
SNDDATA 11000 B000000 9700 F000000
Field References:
FIELD ATTR REFERENCES (M=MODIFIED D=DEFINED)
*IN15 A(1) 4200
*IN30 A(1) 10900M
CHKTRN TAG 3900 41000
DBDATA A(80) 16000 5600
END TAG 6800 85000
ENDSER BEGSR 5400 108000
EOFPSW A(1) 5500 110000
FEEDBK DS(404) 1100 18000
FMTNM A(8) 19000 12700
MAJCOD A(2) 22000 3800 3900 9800 11200
12300
* 7031 MAJMIN A(4) 21000
MINCOD A(2) 23000 12500
NOTOK EXCPT 7700 12000
NOTOKR TAG 3800 76000 9800 11200
OKEND EXCPT 6700 11700
PGMDEV A(10) 20000 12900
PTREC EXCPT 4000 11500
RCVFLD A(80) A000001D 11600
RECDTA TAG 36000 4200
SENDDTA TAG 52000 5800
SNDFLD A(80) B000001D 5600M F000001D
WCFRTN BEGSR 5700 96000
'03' LITERAL 3800 3900 9800 11200
'1' LITERAL 4200 5500 10900 11000
Indicator References:
INDICATOR REFERENCES (M=MODIFIED D=DEFINED)
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RTDBAT 10/03/90 14:54:38 Page 7
*IN 4200 10900M
LR 8600M
OA 14000 12901
15 4200
30 10900M

```

Figure 11-11 (Part 3 of 4). Target Program Example — RTDBAT (User-Defined Formats)

```

* 7031 35
* 7031 45
* 7031 50
* 7031 80          1500M
98                3700M  5300M  5400
***** END OF CROSS REFERENCE *****
5738R61 V2R1M0 910524      IBM AS/400 RPG/400      ICFLIB/RTDBAT      10/03/90 14:54:38      Page      8
      Message Summary
* QRG6103 Severity: 00 Number: 1
      Message . . . . : No Overflow Indicator is specified but an
      indicator is assigned to a file and automatic skip to 6 is
      generated.
* QRG7031 Severity: 00 Number: 5
      Message . . . . : The Name or indicator is not referenced.
* QRG7086 Severity: 00 Number: 1
      Message . . . . : The RPG handles blocking function for file.
      INFDS contents updated only when blocks of data transferred.
* QRG7089 Severity: 00 Number: 1
      Message . . . . : The RPG provides Separate-Indicator area for
      file.
***** END OF MESSAGE SUMMARY *****
5738R61 V2R1M0 910524      IBM AS/400 RPG/400      ICFLIB/RTDBAT      10/03/90 14:54:38      Page      9
      Final Summary
Message Count: (by Severity Number)
      TOTAL  00  10  20  30  40  50
           8   8   0   0   0   0
Program Source Totals:
Records . . . . . : 129
Specifications . . . . . : 59
Table Records . . . . . : 0
Comments . . . . . : 70
PRM has been called.
Program RTDBAT is placed in library ICFLIB. 00 highest Error-Severity-Code.
***** END OF COMPILATION *****

```

Figure 11-11 (Part 4 of 4). Target Program Example — RTDBAT (User-Defined Formats)

```

5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFBAT          10/03/90 14:56:00      Page      1
Compiler . . . . . : IBM AS/400 RPG/400
Command Options:
Program . . . . . : ICFLIB/RTFBAT
Source file . . . . . : ICFLIB/QICFPUB
Source member . . . . . : *PGM
Text not available for message RXT0073 file QRPMSG.
Generation options . . . . . : *NOLIST *NOXREF *NOATR *NODUMP *NOOPTIMIZE
Source listing indentation . . . . . : *NONE
SAA flagging . . . . . : *NOFLAG
Generation severity level . . . . . : 9
Print file . . . . . : *LIBL/QSYSPRT
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Text . . . . . : *SRCMBRTXT
Phase trace . . . . . : *NO
Intermediate text dump . . . . . : *NONE
Snap dump . . . . . : *NONE
Codelist . . . . . : *NONE
Ignore decimal data error . . . . . : *NO

```

```

Actual Program Source:
Member . . . . . : RTFBAT
File . . . . . : QICFPUB
Library . . . . . : ICFLIB
Last Change . . . . . : 03/20/89 15:15:51
Description . . . . . : rpg batch file transfer using $$FORMAT
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFBAT          10/03/90 14:56:00      Page      2

```

SEQUENCE NUMBER	INDDO	DO	LAST	PAGE	PROGRAM
...1...+...2...+...3...+...4...+...5...+...6...+...7...	USE	NUM	UPDATE	LINE	ID
Source Listing					
100	H*		10/16/87		
200	H*		03/20/89		
300	H*		03/20/89		
400	H*		03/20/89		
500	H*		03/20/89		
600	H*		03/20/89		
700	H*		03/20/89		
800	H*		03/20/89		
900	H*		10/16/87		
1000	H* 1		10/16/87		
1100	FTGTICF	CF F 84	10/16/87		WORKSTN
1200	F		10/16/87		KINFDS FEEDBK
1300	FDBFILE	IF F 80	10/16/87		DISK
1400	FQPRINT	O F 132	10/16/87		PRINTER
1500	ITGTICF	NS 80	10/16/87		
1600	I		10/16/87		1 80 RCVFLD
1700	IDBFILE	NS 80	10/16/87		
1800	I		10/16/87		1 80 DBDATA
1900	I*		10/16/87		
2000	I* 2		10/16/87		
2100	IFEEDBK	DS	10/16/87		
2200	I		10/16/87		38 45 FMTNM
2300	I		10/16/87		273 282 PGMDEV
2400	I		10/16/87		401 404 MAJMIN
2500	I		10/16/87		401 402 MAJCOD
2600	I		10/16/87		403 404 MINCOD
2700	C*		10/16/87		
2800	C*		03/20/89		
2900	C*		03/20/89		
3000	C*		03/20/89		
3100	C*		03/20/89		
3200	C*		03/20/89		
3300	C*		10/16/87		
3400	C*		10/16/87		
3500	C*		10/16/87		
3600	C*		03/20/89		

Figure 11-12 (Part 1 of 4). Target Program Example — RTFBAT (System-Supplied Formats)

```

3700 C*****
3800 C* 3
3900 C RECDTA TAG
4000 C READ TGTICF 98 3
4100 C MAJCOD CABGT'03' NOTOKR ERROR?
4200 C MAJCOD CABEQ'03' CHKTRN NO DATA ?
4300 C EXCPTPTREC
4400 C CHKTRN TAG
5738RG1 V2R1M0 910524 IBM AS/400 RP6/400 ICLIB/RTFBAT 10/03/90 14:56:00 Page 3
SEQUENCE NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE IND DO LAST PAGE PROGRAM
4500 C MINCOD CABNE'00' RECDTA RCVTRNRND ? 10/16/87
4600 C*****
4700 C* 03/20/89
4800 C* WHEN A RCVTRNRND INDICATION IS RECEIVED, THE PROGRAM STARTS 10/16/87
4900 C* SENDING RECORDS TO THE SOURCE PROGRAM. RECORDS ARE SENT UNTIL 10/16/87
5000 C* THE END OF FILE IS REACHED ON THE DATABASE FILE. AT THIS TIME 03/20/89
5100 C* A DETACH REQUEST IS SENT TO THE SOURCE PROGRAM. 03/20/89
5200 C* 03/20/89
5300 C***** 10/16/87
5400 C* 4
5500 C SENDTA TAG
5600 C READ DBFILE 98 3
5700 C 98 EXSR ENDSSES SEND DETACH 10/16/87
5800 C EOFPSW IFNE '1' B001 10/16/87
5900 C EXSR WCFRTN 001 10/16/87
6000 C GOTO SENDTA SEND DATA 001 10/16/87
6100 C END E001 10/16/87
6200 C***** 10/16/87
6300 C* 03/20/89
6400 C* WHEN THE END OF FILE IS REACHED, AN EOJ MESSAGE IS 10/16/87
6500 C* PRINTED, AND CONTROL GOES TO END. 03/20/89
6600 C* 03/20/89
6700 C***** 10/16/87
6800 C* 5
6900 C EXCPTOKEND 10/16/87
7000 C GOTO END 10/16/87
7100 C***** 10/16/87
7200 C* 03/20/89
7300 C* WHEN AN I/O OPERATION ERROR IS DETECTED, AN ABNORMAL 10/16/87
7400 C* TERMINATION MESSAGE IS PRINTED AND THE PROGRAM ENDS. 10/16/87
7500 C* 03/20/89
7600 C***** 10/16/87
7700 C* 6
7800 C NOTOKR TAG 10/16/87
7900 C EXCPTNOTOK 10/16/87
8000 C***** 10/16/87
8100 C* 03/20/89
8200 C* WHEN PROCESSING IS FINISHED, THE LAST RECORD SWITCH IS SET 10/16/87
8300 C* AND THE PROGRAM IS ENDED. 10/16/87
8400 C* 03/20/89
8500 C***** 10/16/87
8600 C* 7
8700 C END TAG 10/16/87
8800 C SETON LR 1 10/16/87
8900 C RETRN 10/16/87
9000 C***** 10/16/87
9100 C* 03/20/89
9200 C* THIS SUBROUTINE IS CALLED TO SEND DATA TO THE SOURCE PROGRAM. 03/20/89
9300 C* IF A SESSION ERROR OCCURS, AN ABNORMAL TERMINATION MESSAGE IS 03/20/89
9400 C* PRINTED, THE LR SWITCH IS SET, AND THE JOB IS ENDED. 03/20/89
9500 C* 03/20/89
9600 C***** 10/16/87

```

Figure 11-12 (Part 2 of 4). Target Program Example — RTFBAT (System-Supplied Formats)


```

9700 C* 8 10/16/87
9800 C WCFRTN BEGSR 10/16/87
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RTFBAT 10/03/90 14:56:00 Page 4
SEQUENCE IND DO
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE NUM UPDATE LINE ID
9900 C EXCPTSNDATA 10/16/87
10000 C MAJCOD CABGT'03' NOTOKR ERROR? 10/16/87
10100 C ENDSR 10/16/87
10200 C***** 10/16/87
10300 C* 03/20/89
10400 C* THIS SUBROUTINE IS CALLED AT END OF FILE TO SEND AN 10/16/87
10500 C* INDICATION TO THE LOCAL SYSTEM THAT TRANSMISSION IS ENDED. 10/16/87
10600 C* THE END OF FILE SWITCH IS SET TO END THE JOB. 03/20/89
10700 C* 03/20/89
10800 C***** 10/16/87
10900 C* 9 10/16/87
11000 C ENDSER BEGSR 10/16/87
11100 C MOVE '1' *IN30 ACTV DETACH 10/16/87
11200 C MOVE '1' EOFPSW 1 10/16/87
11300 C EXCPTENDREC SEND DETACH 10/16/87
11400 C MAJCOD CABGT'03' NOTOKR ERROR? 10/16/87
11500 C ENDSR 10/16/87
11600 C***** 10/16/87
11700 OQPRINT E 1 PTREC 10/16/87
11800 0 RCVFLD 80 10/16/87
11900 0 E 1 OKEND 10/16/87
12000 0 21 'RTFBAT HAS COMPLETED ' 10/16/87
12100 0 30 'NORMALLY.' 10/16/87
12200 0 E 1 NOTOK 10/16/87
12300 0 21 'RTFBAT HAS COMPLETED ' 10/16/87
12400 0 32 'ABNORMALLY.' 10/16/87
12500 0 MAJCOD 37 10/16/87
12600 0 38 '/' 10/16/87
12700 0 MINCOD 40 10/16/87
12800 0 49 'FORMAT:' 10/16/87
12900 0 FMTNM 60 10/16/87
13000 0 69 'DEVICE:' 10/16/87
13100 0 PGMDEV 80 03/20/89
13200 OTGTICF E SNDATA 10/16/87
13300 0 K8 '$$SENDNI' 10/16/87
13400 0 4 '0000' 10/16/87
13500 0 DBDATA 84 10/16/87
13600 0 E ENDREC 10/16/87
13700 0 K8 '$$SENDER' 10/16/87
13800 0 4 '0000' 10/16/87
* 6103 13801 OVERFLOW INDICATOR 0A ASSIGNED TO FILE QPRINT.
***** END OF SOURCE *****
Additional Diagnostic Messages
* 7089 1100 RPG PROVIDES SEPARATE INDICATOR AREA FOR FILE TGTCF.
* 7086 1300 RPG PROVIDES BLOCK OR UNBLOCK SUPPORT FOR FILE DBFILE.
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RTFBAT 10/03/90 14:56:00 Page 5
Cross Reference
File and Record References:
FILE/RCD DEV/RCD REFERENCES (D=DEFINED)
02 DBFILE DISK 13000 1700 5600
03 QPRINT PRINTER 14000 11700 11900 12200 13801
01 TGTCF WORKSTN 11000 1500 4000 13200 13600
$$SENDER 13700
$$SENDNI 13300
Field References:
FIELD ATTR REFERENCES (M=MODIFIED D=DEFINED)
*IN30 A(1) 11100M
CHKTRN TAG 4200 4400D
DBDATA A(80) 1800D 1350D
END TAG 7000 8700D
ENDREC EXCPT 11300 1360D
ENDSES BEGSR 5700 11000D
EOFPSW A(1) 5800 11200D
FEEDBK DS(404) 1100 2100D
FMTNM A(8) 2200D 1290D

```

Figure 11-12 (Part 3 of 4). Target Program Example — RTFBAT (System-Supplied Formats)

```

MAJCOD      A(2)    25000 4100  4200 10000 11400 12500
* 7031 MAJMIN      A(4)    24000
MINCOD      A(2)    26000 4500 12700
NOTOK       EXCPT   7900  12200
NOTOKR      TAG     4100  78000 10000 11400
OKEND       EXCPT   6900  11900
PGMDEV      A(10)   23000 13100
PTREC       EXCPT   4300  11700
RCVFLD      A(80)   16000 11800
RECDTA      TAG     39000 4500
SENDTA      TAG     55000 6000
SNDATA      EXCPT   9900  13200
WCFRTN      BEGSR   5900  98000
'00'        LITERAL 4500
'03'        LITERAL 4100  4200 10000 11400
'1'         LITERAL 5800  11100 11200

Indicator References:
INDICATOR REFERENCES (M=MODIFIED D=DEFINED)
*IN         11100M
LR          8800M
OA          1400D 1380I

5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFBAT          10/03/90 14:56:00          Page          6
30          11100M
* 7031 80          1500M 1700M
98          4000M 5600M 5700
***** END OF CROSS REFERENCE *****
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFBAT          10/03/90 14:56:00          Page          7
          M e s s a g e S u m m a r y
* QR66103 Severity: 00 Number: 1
  Message . . . . : No Overflow Indicator is specified but an
                    indicator is assigned to a file and automatic skip to 6 is
                    generated.
* QR67031 Severity: 00 Number: 2
  Message . . . . : The Name or indicator is not referenced.
* QR67086 Severity: 00 Number: 1
  Message . . . . : The RPG handles blocking function for file.
                    INFDS contents updated only when blocks of data transferred.
* QR67089 Severity: 00 Number: 1
  Message . . . . : The RPG provides Separate-Indicator area for
                    file.
          ***** END OF MESSAGE SUMMARY *****
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFBAT          10/03/90 14:56:00          Page          8
          F i n a l S u m m a r y
Message Count: (by Severity Number)
TOTAL      00      10      20      30      40      50
           5       5       0       0       0       0

Program Source Totals:
Records . . . . . : 138
Specifications . . . . . : 67
Table Records . . . . . : 0
Comments . . . . . : 71
PRM has been called.
Program RTFBAT is placed in library ICFLIB. 00 highest Error-Severity-Code.
***** END OF COMPILATION *****

```

Figure 11-12 (Part 4 of 4). Target Program Example — RTFBAT (System-Supplied Formats)

Multiple-Session Inquiry (Example II)

This example illustrates an interactive inquiry application that communicates with multiple ICF sessions. A source AS/400 system program accepts inquiries from a display device and sends a request to one of four AS/400 systems. The source program communicates with the display device through a display file, and with the four remote systems through a single ICF file.

The purpose of this example is to show multiple sessions from a single ICF file. The source

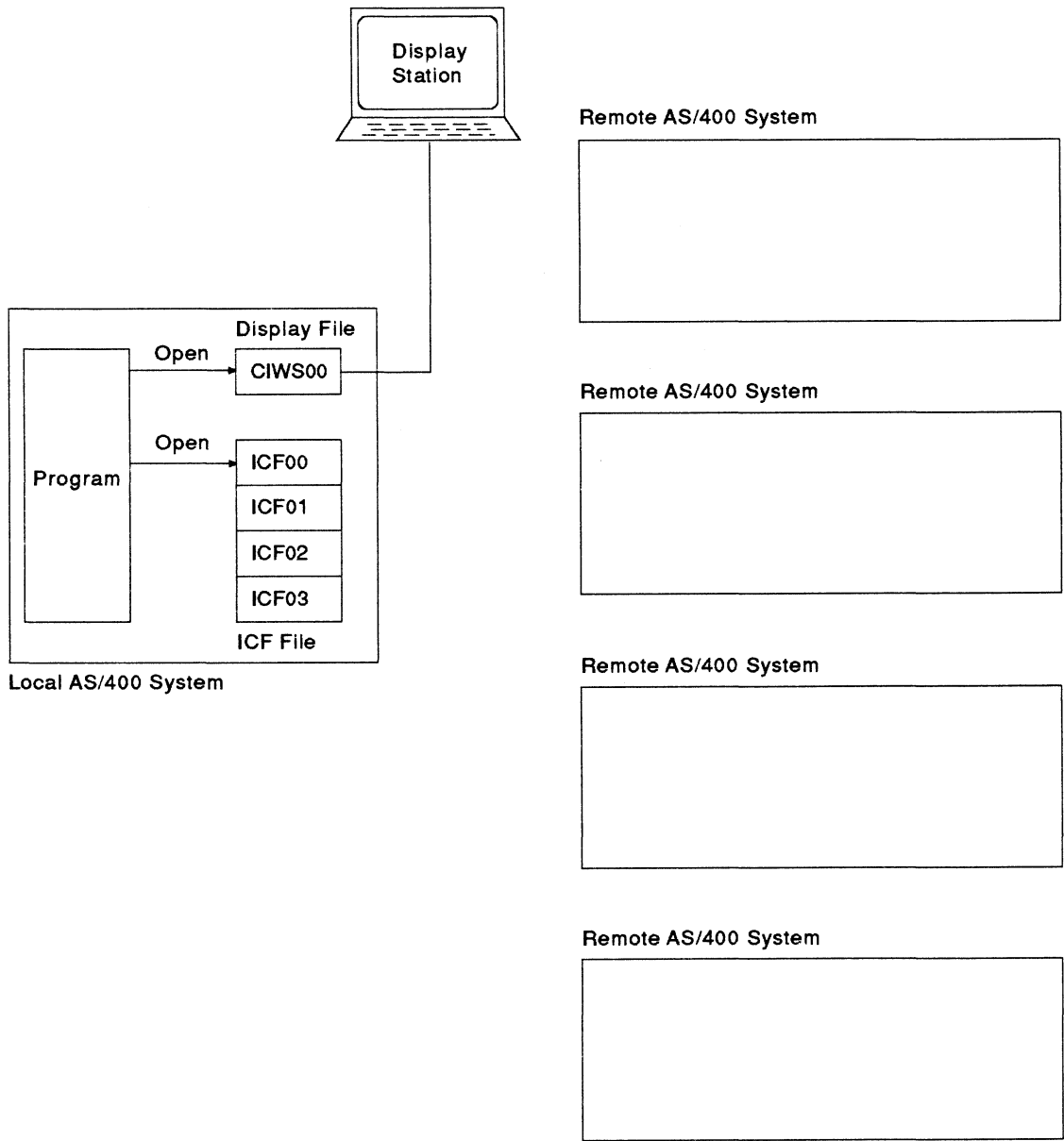
program communicates with four sessions. From the viewpoint of each of the four target programs, the requester is the only session. Therefore, the target programs do not require any unique logic to support the multiple-session source.

Both the source program and the target program are described. The same target program is evoked in each of the four separate remote systems. Therefore, only one target program is shown in the programming example.

Transaction Flow of the Multiple-Session Inquiry (Example II):

The program shown in Figure 11-13 is started from a display station. Both the display and the ICF files are opened. CIWS00 is the *REQUESTER device, and is acquired when the display file opens. CIWS00 is acquired because DEV(*REQUESTER) was speci-

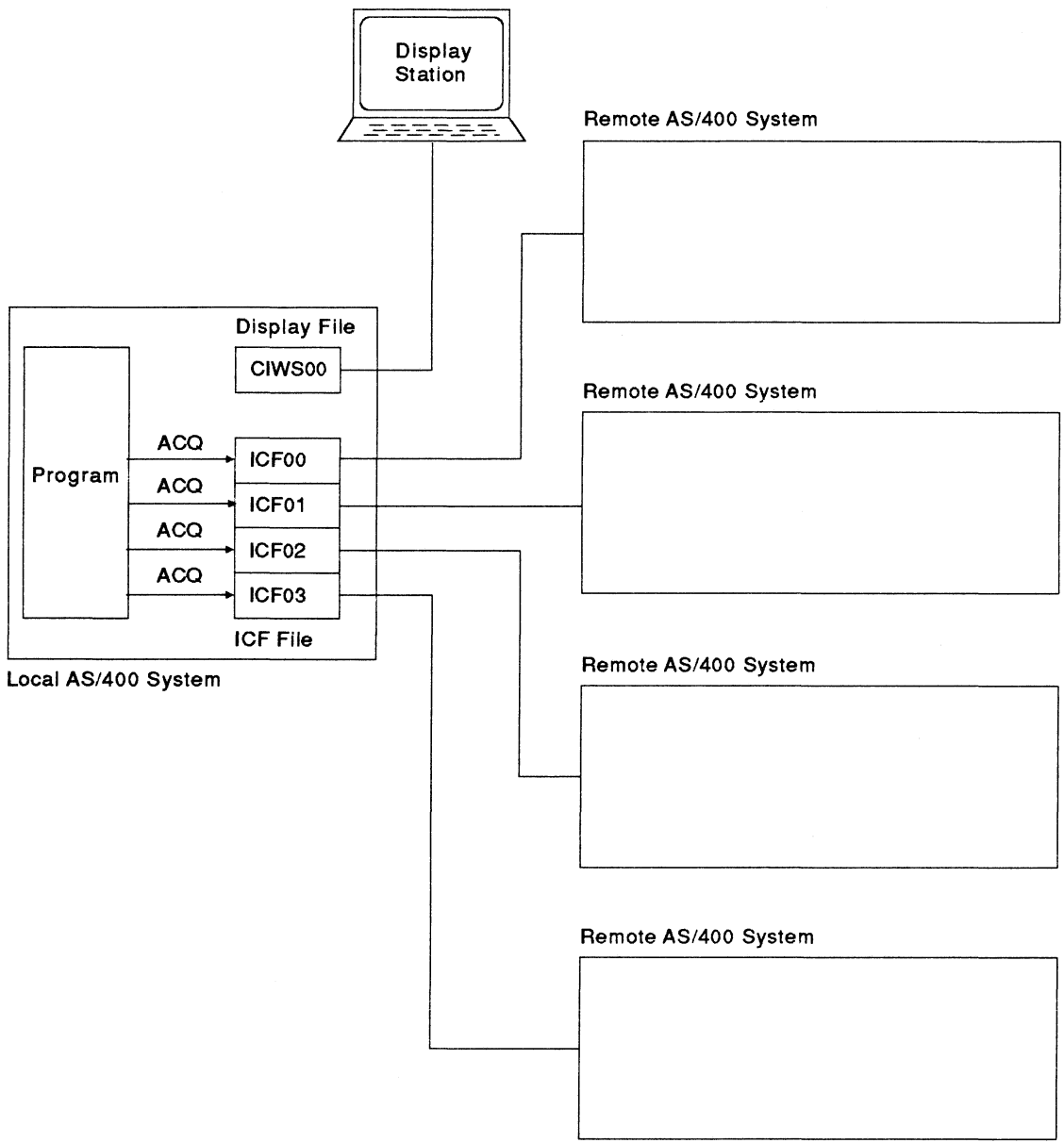
fied when the display file was created. Since the ICF file was created with ACQPGMDEV(*NONE), no ICF devices are acquired during open processing.



RSL100-4

Figure 11-13. Program Starts at Display Station

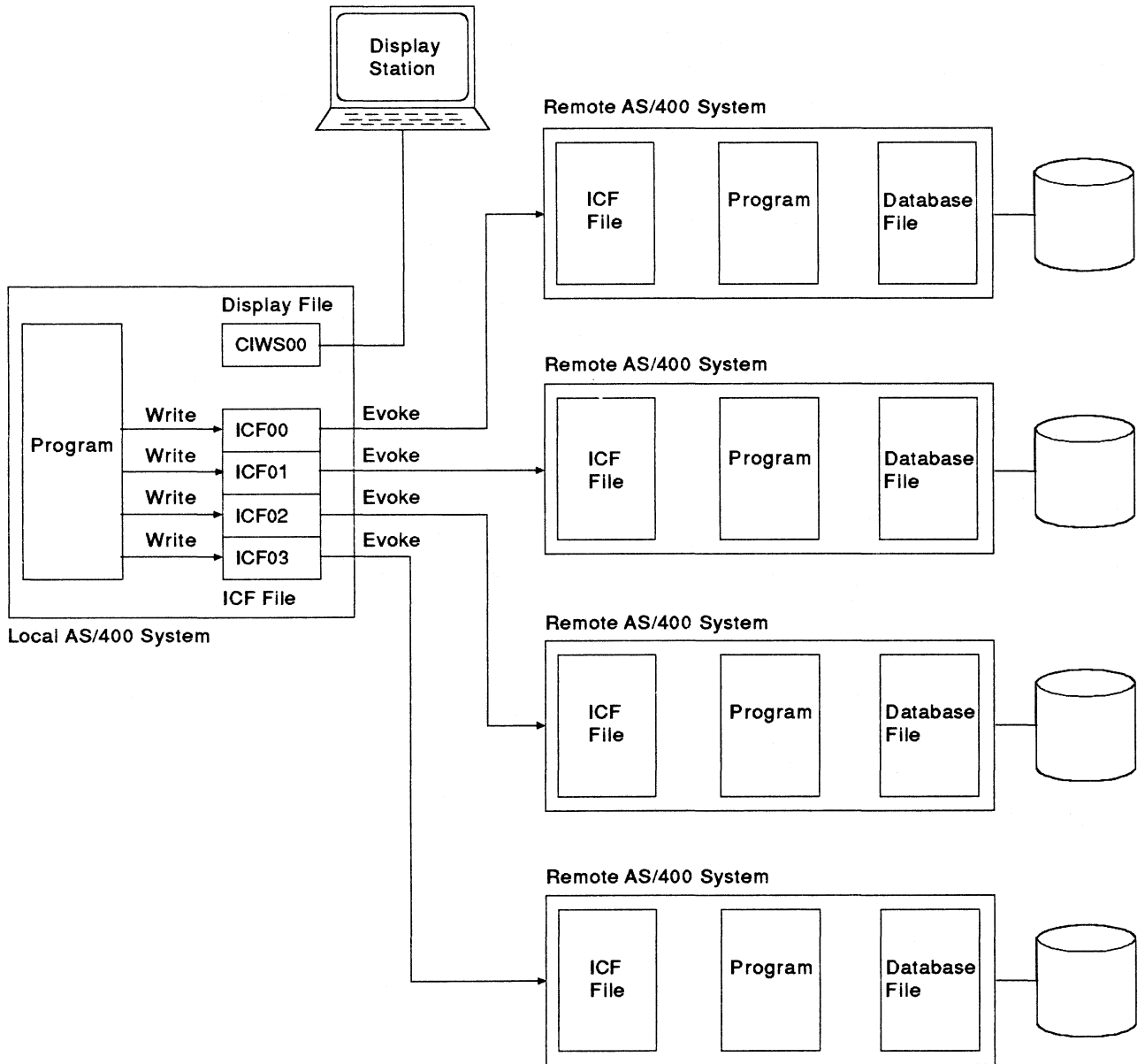
All other program devices must be explicitly acquired by the program, as shown in Figure 11-14.



RSL5651-4

Figure 11-14. Program Devices Explicitly Acquired

All target programs are started with an evoke,
as shown in Figure 11-15.

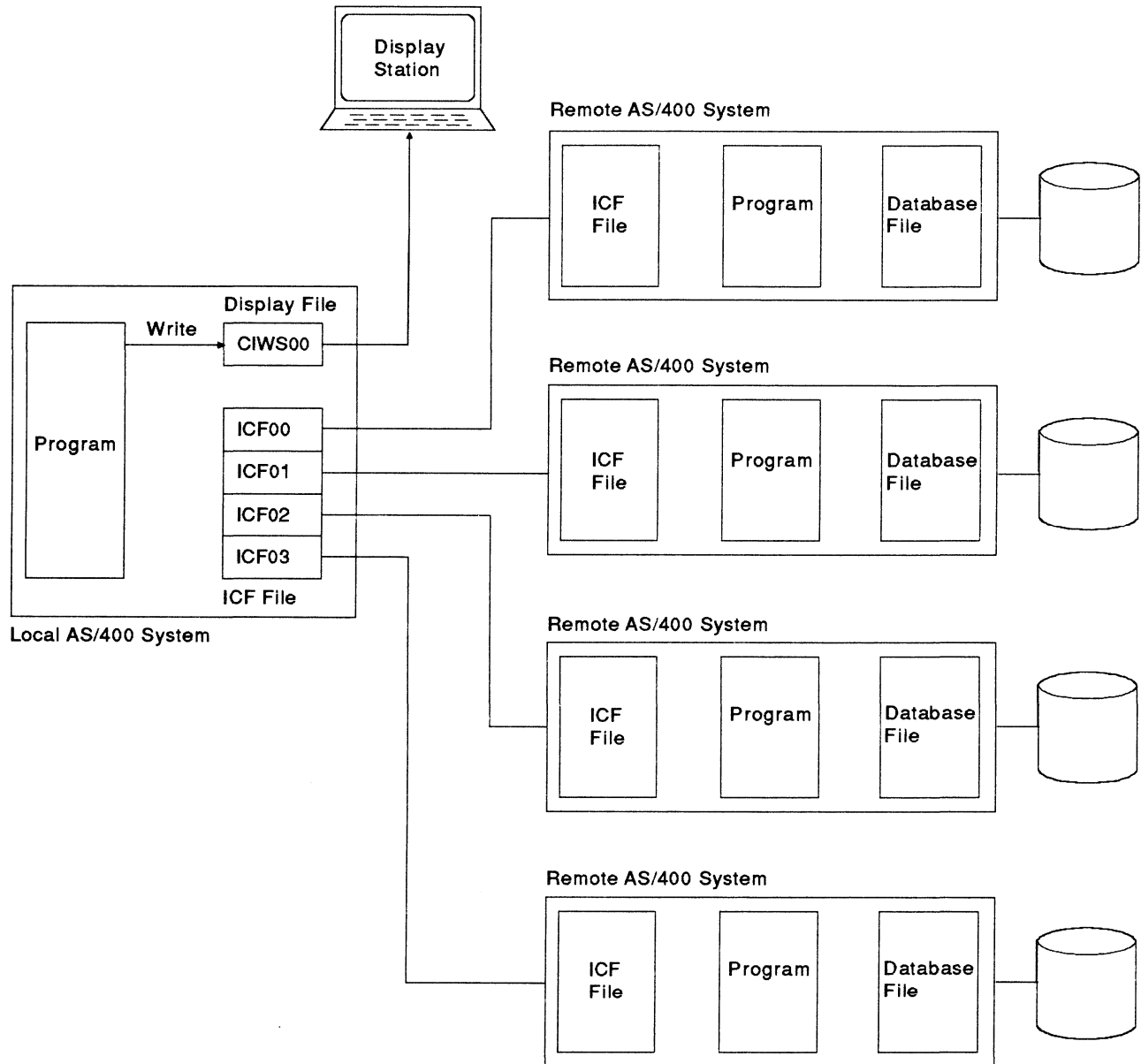


RSL662-4

Figure 11-15. Evoke Starts Target Programs

The source program uses a specific program device name. Each target program uses an ICF file with a program device name that is associated with the requester. The target program's only session is the one used to communicate with the source program. The ICF file is implicitly opened by the RPG/400 language support when the target program is started. Since the

file was created with the requesting program device specified on the ACQPGMDEV parameter, the requesting program device is acquired with the implicit open. The main menu is written to the display station on the local system and the program waits for a request from the display station, as shown in Figure 11-16.

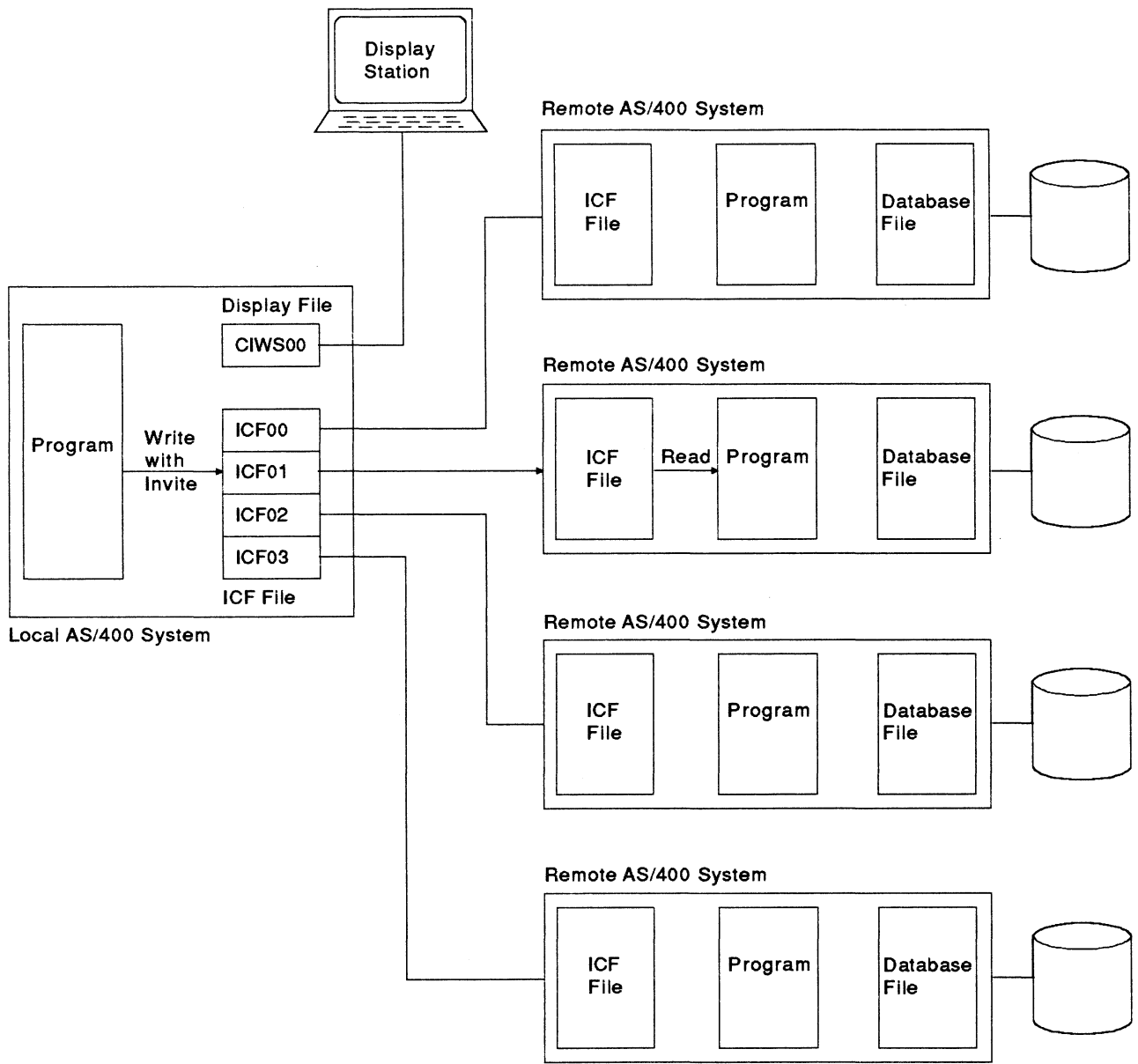


RSL5653-5

Figure 11-16. Main Menu Written to Display Station

The source program sends an inquiry request to one of the remote systems based on the request

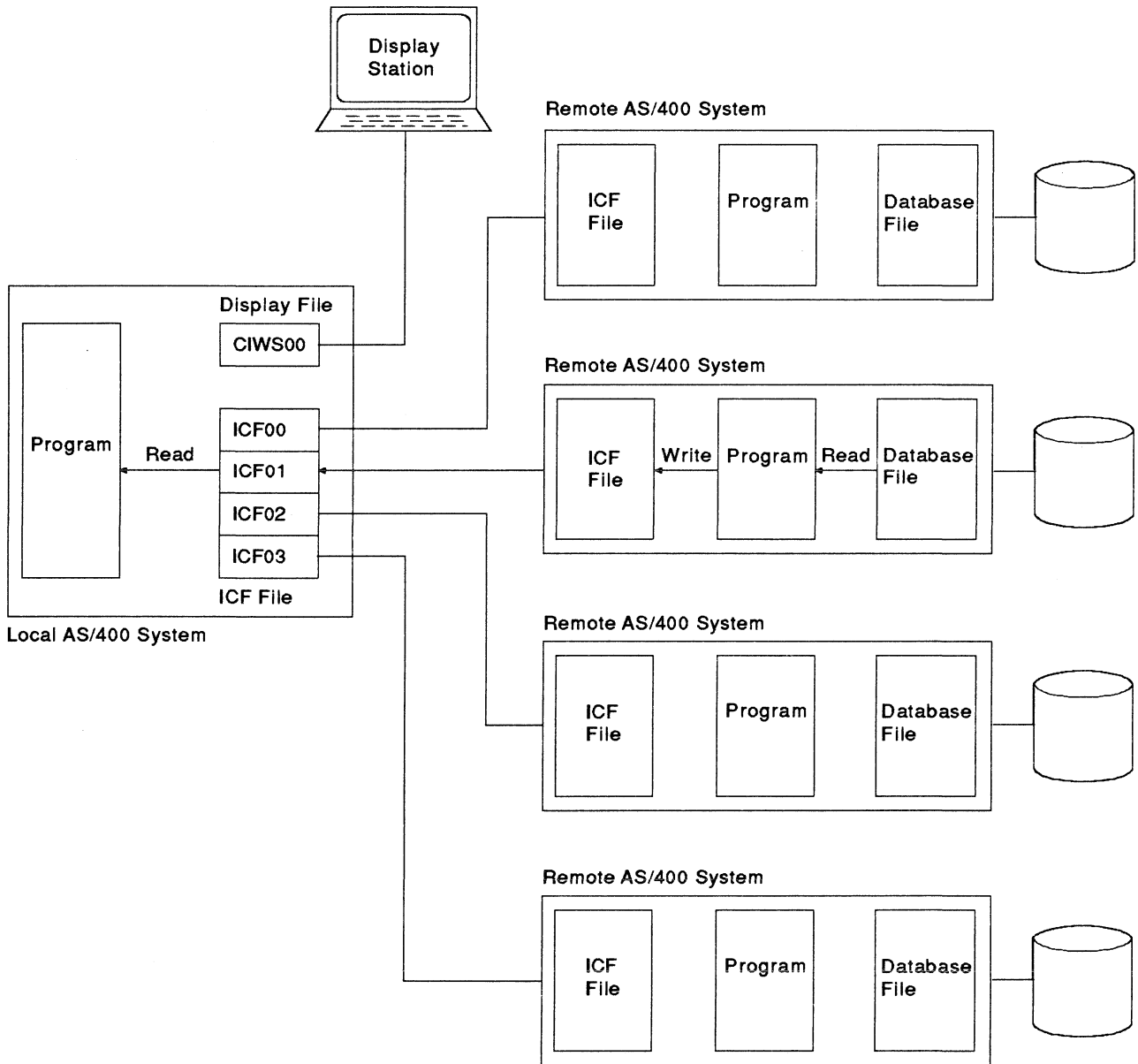
made from the display station, as shown in Figure 11-17.



RSL654-4

Figure 11-17. Program Sends Inquiry Request to Remote System

The target program responds to the inquiry by sending a reply, as shown in Figure 11-18.

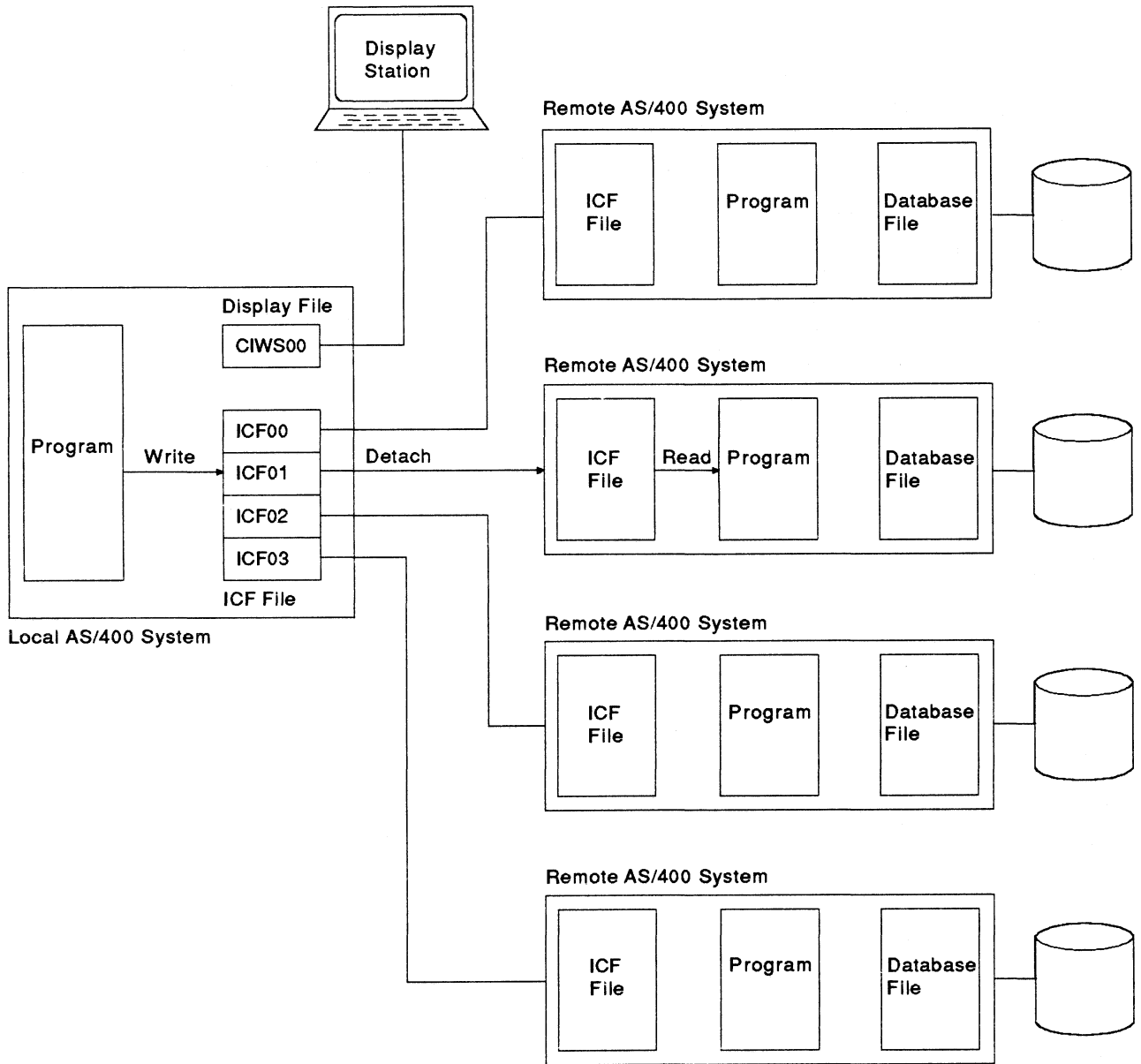


RSL655-4

Figure 11-18. Target Program Sends a Reply

The program sends a detach request and ends the session when command function key 1 is

pressed (while the main inquiry menu is present), as shown in Figure 11-19.



RSL5656-5

Figure 11-19. Program Ends the Session

Source Program Multiple-Session

Inquiry (Example II): The following describes a source program multiple-session inquiry.

Program Files: The RPG/400 multiple-session source program uses the following files:

CMNFIL

A ICF file used to send records to and receive records from the target program.

DSPFIL

A display file used to enter requests to be sent to the target program.

QPRINT

A printer file used to print error messages resulting from communications errors.

DDS Source: The DDS for the ICF file (CMNFIL) is illustrated below.

```

5714PW1 R01M00 910524          SEU SOURCE LISTING          10/14/87 17:20:41          PAGE 1
SOURCE FILE . . . . . QICFPUB/ICFLIB
MEMBER . . . . . CMNFIL
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100
200
300  A*****
400  A*
500  A*          ICF FILE          *
600  A*          USED IN SOURCE MULTIPLE SESSION PROGRAM      *
700  A*
800  A*****
900  A          INDARA
1000 A          R ITMRSP
1100 A          RECID(1 'I')
1200 A          RECITM          1
1300 A          ITEMNO          6 0
1400 A          DESC          30
1500 A          QTYLST          7 0
1600 A          QTYOH          7 0
1700 A          QTYOO          7 0
1800 A          QTYBO          7 0
1900 A          UNITQ          2
2000 A          PR01          7 2
2100 A          PR05          7 0
2200 A          UFRT          5 2
2300 A          SLSTM          9 2
2400 A          SLSTY          11 2
2500 A          CSTTM          9 2
2600 A          CSTTY          11 2
2700 A          PRO          5 2
2800 A          LOS          9 2
2900 A          FILL1          56
3000 A          R DTLRSP
3100 A          RECID(1 'C')
3200 A          RCVTRNRD(90)
3300 A          RECCUS          1
3400 A          CUSTNO          6 0
3500 A          DNAME          30
3600 A          DLSTOR          6 0
3700 A          DSLSTM          9 0
3800 A          DSPM01          9 0
3900 A          DSPM02          9 0
4000 A          DSPM03          9 0
4100 A          DSTTYD          11 0
4200 A          IDEPT          3 0
4300 A          FILL2          57
4400 A          R DETACH
4500 A          DETACH
4600 A          R EOS
4700 A          EOS
4800 A          R EVKREQ
4900 A          EVOKE(&LIB/&PGMID)
5000 A          PGMID          10A P
5100 A          LIB          10A P
5200 A          R ITMREQ
5300 A          INVITE
5400 A          ITEMNO          6 0
5500 A          R DTLREQ
5600 A          INVITE
5700 A          CUSTNO          6 0
          *****
          * * * * * E N D   O F   S O U R C E   * * * * *

```

The DDS source file for the display file (DSPFIL) is shown below.

```

000100071007 A*****
000200071007 A*
000300071007 A*          DISPLAY FILE          *
000400071007 A*          USED IN SOURCE MULTIPLE SESSION PROGRAM *
000500071007 A*
000600071007 A*****
000700071008 A* BEGINNING MENU
000800071008 A*****
000900071007 A          DSPSIZ(*DS3)
001000071007 A          CF01(99) CF02(98) CF03(97)
001100071007 A          TEXT('MENU FOR INQUIRY')
001200071007 A          R CIMENU
001300071007 A          1 34'INQUIRY MENU'
001400071007 A          3 1'Select one of the following:'
001500071007 A          4 3'1. Item inquiry'
001600071007 A          5 3'2. Customer inquiry'
001700071007 A          11 1'Option:'
001800071007 A          OPTION          1N 1 11 9VALUES('1' '2')
001900071007 A          19 5DFT('CMD KEY 1 - END ')
002000071007 A          R DTLMNU          TEXT('CUSTOMER INQUIRY SCREEN 1')
002100071013 A          CUSTNO          6N 01 2 20
002200071008 A          19 5DFT('CMD KEY 1 - END ')
002300071008 A          19 23DFT(' 2 - MAIN MENU ')
002400071008 A*
002500071008 A*****
002600071007 A* CUSTOMER INQUIRY SCREEN
002700071008 A*****
002800071007 A          R DTLSRC          TEXT('CUSTOMER INQUIRY SCR. #2')
002900071007 A          1 3DFT('CUST DPT LAST ORD & THIS
003000071007 A          $MTH1 &MTH2 $MTH3
003100071008 A          THIS YTD NAME')
003200071008 A          CUSTN          6N 2 2
003300071007 A          DEPT          3N 0 2 9
003400071007 A          DLSTR          6N 0 2 13
003500071007 A          DSLSM          9N 0 2 22
003600071007 A          DSPM1          9N 0 2 32
003700071007 A          DSPM2          9N 0 2 42
003800071007 A          DSPM3          9N 0 2 52
003900071007 A          DSTYD          11N 0 2 62
004000090321 A          CNAME          5 2 74
004100071008 A          19 5DFT('CMD KEY 1 - END ')
004200071008 A          19 23DFT(' 2 - MAIN MENU ')
004300071007 A*
004400071008 A*****
004500071007 A* ITEM INQUIRY SCREEN
004600071008 A*****
004700071007 A          R ITMNU          TEXT('ITEM INQUIRY SCREEN ONE')
004800071008 A          2 2DFT('ENTER ITEM NUMBER')
004900071013 A          ITEMNO          6N 01 2 20
005000071008 A          19 5DFT('CMD KEY 1 - END ')
005100071008 A          19 23DFT(' 2 - MAIN MENU ')
005200071008 A*
005300071008 A* ITEM DISPLAY
005400071008 A*****
005500071007 A          R ITMSC2          TEXT('ITEM INQUIRY SCREEN TWO')
005600071007 A          OVERLAY
005700071007 A          4 2DFT('DESC-')
005800071007 A          DSC          30 4 8
005900071007 A          5 2DFT('QUANTITY AVAILABLE')
006000071007 A          QAVAIL          7N 0 5 25
006100071007 A          6 11DFT('ON HAND')
006200071007 A          QTYH          7N 0 6 25
006300071007 A          7 11DFT('ON ORDER')
006400071007 A          QTYO          7N 0 7 25
006500071007 A          8 11DFT('BACK ORDER')
006600071007 A          QTYB          7N 0 8 25
006700071007 A          9 2DFT('UNIT OF MEASURE')
006800071007 A          UNT          2 9 30
006900071007 A          10 2DFT('PRICE PER UNIT')
007000071007 A          PRI          7Y 2 10 24EDTCDE(3)
007100071007 A          11 8DFT('QUANTITY')
007200071007 A          PRS          7Y 0 11 25EDTCDE(3)
007300071007 A          12 8DFT('FREIGHT')
007400071007 A          UFR          5Y 2 12 26EDTCDE(3)
007500071008 A          13 32DFT('MORE... ')
007600071008 A          19 5DFT('CMD KEY 1 - END ')
007700071008 A          19 23DFT(' 2 - MAIN MENU ')
007800071008 A          19 40DFT(' 3 - ITEM MENU ')

```

```

007900071008 A*****
008000071008 A* ITEM ADDITIONAL DISPLAY
008100071008 A*****
008200071007 A          R ITMSC3          TEXT('ITEM INQUIRY SCREEN 3 ')
008300071007 A          OVERLAY
008400071007 A          5 2DFT('SALES MONTH')
008500071007 A          SLSM          9Y 2 5 16EDTCDE(1)
008600071007 A          6 8DFT('Y-T-D')
008700071007 A          SLSY          11Y 2 6 14EDTCDE(1)
008800071007 A          7 2DFT('COSTS MONTH')
008900071007 A          CSTM          9Y 2 7 16EDTCDE(1)
009000071007 A          8 8DFT('Y-T-D')
009100071007 A          CSTY          11Y 2 8 14EDTCDE(1)
009200071007 A          9 2DFT('PROFIT PCT')
009300071007 A          PROFIT          5Y 2 9 22EDTCDE(1)
009400071007 A          LOSTS          9Y 2 10 16EDTCDE(1)
009500071007 A          19 5DFT('CMD KEY 1 - END ')
009600071008 A          19 23DFT(' 2 - MAIN MENU ')
009700071008 A*
009800071008 A* TIMOUT SCREEN
009900071008 A*****
010000071007 A          R TIMOUT          TEXT('TIME OUT SCREEN')
010100071007 A          OVERLAY
010200071007 A          20 2DFT('REMOTE SYSTEM TIMED OUT. ENTER
010300071007 A          1 TO TRY AGAIN OR 2 TO END.')
010400071007 A          TIMRSP          1 1 20 61
010500071007 A

```

ICF File Creation and Program Device Entry

Definition: The command needed to create the ICF file is:

```

CRTRICFF FILE(ICFLIB/CMNFIL) SRCFILE(ICFLIB/QICFPUB)
SRCMBR(CMNFIL) ACQPGHDEV(*NONE) MAXPGHDEV(4) WAITRCD(30)
TEXT("SOURCE ICF FILE FOR MULTIPLE SESSION PROGRAM")

```

The commands needed to define the four program device entries are:

```

OVRICFDEVE PGMDEV(ICF00) RMTLOCNAME(CHICAGO) FMTSLT(*RECID)
OVRICFDEVE PGMDEV(ICF01) RMTLOCNAME(NEWYORK) FMTSLT(*RECID)
OVRICFDEVE PGMDEV(ICF02) RMTLOCNAME(DETROIT) FMTSLT(*RECID)
OVRICFDEVE PGMDEV(ICF03) RMTLOCNAME(MADISON) FMTSLT(*RECID)

```

Program Explanation: The following explains the structure of the program examples illustrated in Figure 11-20 on page 11-38 and in Figure 11-21 on page 11-51. The ICF file used in the first example is defined by the user, and uses externally described data formats (DDS). The second example uses the same file, but uses program-described data and system-supplied formats. The reference numbers in the explanation below correspond to the numbers in the following program examples.

Although the basic structure of the two examples provided is the same, there are differences because of the way the user-defined formats and the system-supplied formats are used. All output operations to the ICF file in the first example are done using the WRITE statement. All output operations in the ICF file in the second example using system-supplied formats are done using the EXCPT statement.

Differences between the first and second example are described as notes in each of the following descriptions where necessary.

- 1** The file specifications define the ICF file (CMNFIL) and the display file (DSPFIL) used in the program.

CMNFIL is the ICF file used to send records to and receive records from each of the four target programs.

DSPFIL is the display file used to receive user's requests and to report the information received based on the request.

The files used in the program are opened at the beginning of the RPG/400 cycle.

Note: In the program using system-supplied formats, the input records for CMNFIL are explicitly coded in the program since CMNFIL is now treated as a program-described file. The system-supplied file, QICDMF, can be used instead of CMNFIL. To use QICDMF, specify QICDMF in the file specification, or use an OVRICFF command to change the file name from CMNFIL to QICDMF.

The continuation lines on the file specification define the following:

- The data structure names, IOFB and IODS, used for the feedback area (INFDS) for CMNFIL and DSPFIL respectively.
- The number of program devices that can be attached to the files (four for CMNFIL).
- The program device name in *CMID* field to which it issues the I/O operation.

- 2** The file information data structure (IOFB) is provided to receive the I/O feedback area following an ICF file I/O operation.

For the display file, the file information data structure (IODS) is used by the program to determine the record format used for the last display file I/O operation. The field name referenced in the program is *RECID*, found in positions 261 through 268 of the feedback area.

- 3** The four ICF program devices used by the program are explicitly acquired.

The work station is implicitly acquired when the DSPFIL file opens.

Also, the evoke requests are issued to the remote systems by the subroutine at **13**.

When control returns from **13**, the main menu (record format CIMENU) is written to the work station.

- 4** A read operation is issued to the display program device and the program waits for an input request from the user. When a record is returned, the last record format used (as specified in the *RECID* field in the I/O feedback area) is checked. The program branches to the appropriate routine according to the value in *RECID*.

- 5** The request entered by the user from the main menu (CIMENU) is checked. If indicator 99 is set to 1, indicating that the operator pressed function key 1, the four transactions and sessions end and the program ends. If the operator entered option 1, the program writes the item inquiry menu (ITMMNU) to the work station and returns to **4**.

If the option is not 1, the customer inquiry menu (DTLMNU) is written to the work station and control is passed to **4**.

- 6** The item number requested by the user from the Item Inquiry Screen (record format ITMMNU) is processed here. If function key 1 is pressed (indicator 99), control passes to **12**, the four transactions and sessions are ended, and the program ends. If function key 2 is pressed, the inquiry request is canceled, the main menu (CIMENU) is written to the work station, and the program returns to **4**.

The item number read from the work station is checked for value range. If the range is from 0 to 399999, then the request is sent to the target program on program device ICF01.

If the range is from 400000 to 699999, then the request is sent to the target program on program device ICF02.

If the range is from 700000 to 899999, then the request is sent to the target program on program device ICF03.

The request is sent to the appropriate target program by writing data to the program device using format ITMREQ. The *INVITE* keyword is specified as part of the

ITMREQ format to give the target program permission to send.

A read-from-invited-program-devices operation is issued to the invited program device to receive the response to the inquiry. The operation is interpreted as a read-from-invited-program-devices because the program device name field (*CMID*) is blank. Indicator 89 is set on after I/O operation, if the operation does not complete. Subroutine **14** gets control, and further checks are made.

The return codes are checked after every I/O request. If there are any errors, control is passed to **12**.

The program returns to **4**.

Note: In the program using system-supplied formats, the \$\$\$SEND format is used instead of the user-defined ITMREQ format. Also, the EXCPT statement is used instead of the WRITE statement.

- 7** The information received from the target program is processed. If the returned item number is 0 or less, the request is invalid, a new item inquiry menu (ITMMNU) is written to the work station, and control goes to **4**.

The program then performs the calculations to set the quantity fields and writes the result to the requesting work station using record format ITMSC2.

The program then returns to **4**.

- 8** This section processes the user requests for additional information (record format ITMSC2). If function key 2 (indicator 98) was pressed, the main menu (record format CIMENU) writes to the work station and control goes to **4**.

If function key 2 was pressed (as indicated by indicator 98), the profit and loss figures are calculated. Those values are then written to the work station using format ITMSC3 (item inquiry work station 3). The program then returns to **4**. If function key 1 (indicator 99) was pressed, control goes to **12**.

If function key 3 (indicator 97) was pressed, the Item Inquiry Menu (ITMMNU) is written to the work station and the program returns to **4**.

- 9** This section processes requests read from the customer menu (DTLMNU). If function key 2 (indicator 98) was pressed, the main menu (CIMENU) is written to the work station and the program returns to **4**. If function key 1 (indicator 99) was pressed, control goes to **12**.

The customer inquiry request is sent to the target program by writing data to the program device (ICF00) using format DTLREQ. The INVITE keyword is specified as part of the DLTREQ format to give the target program permission to send.

A read operation is issued to the invited program device to receive the response to the inquiry. This is accomplished by blanking out *CMID*. Indicator 88 is set on if the I/O operation did not complete.

The return codes are checked after every I/O request. If there are any errors, control is passed to **12**.

Note: In the program using system-supplied formats, the \$\$\$SEND format is used instead of the user-defined DTLREQ format. Also, the EXCPT operation is used instead of the WRITE operation. The READ operation is issued using ICF file CMNFIL in factor 2.

- 10** The information supplied by the target program in response to a request for a customer detail is processed. If the customer number is 0 or less, the request is invalid and the main menu (record format CIMENU) is written to the work station. The program then returns to **4**.

The detail information is written to the work station using record format DTLSCR.

The program then returns to **4**.

The return codes are checked after every I/O request to verify the success of the operation.

Note: The READ operation is issued using file name CMNFIL in factor 2 in the program using system-supplied formats.

- 11** Control is passed here if the customer detail record format (DLTSCR) is displayed. If function key 1 (indicator 99) was pressed, control goes to **12**. If function key 2 (indicator 98) was pressed, the main menu

(CIMENU) is written to the work station and control is returned to **4**.

- 12** If the record format name is not found on a read operation, an error message prints. If an error occurs on any ICF operation, control is passed here and an error message is printed containing the program device and error that occurred.

For each of the four sessions, the transaction is ended by issuing a detach request to the appropriate program device using format DETACH, and the session is ended by the release operation. The last record indicator is turned on to end the program. The ICF file is implicitly closed at the end of the RPG/400 cycle.

- 13** This subroutine builds the evoke requests to send to the remote systems. Because the DDS keyword for the record format only specifies the field identifiers with the record, this code moves the literal value RTDMULCL to the field *PGMID*, and ICFLIB to the field *LIB*.

When the program start request is received at the remote system, ICFLIB is searched for RTDMULCL and that program is then

started. RTDMULCL is a CL program that contains the following:

```
ADDLIB ICFLIB
CALL ICFLIB/RTDMUL
```

Note: In the program using system-supplied formats, the library and program (ICFLIB/RTFMULCL) are specified as part of the \$\$EVOKNI format. RTFMULCL is a CL program that contains the following:

```
ADDLIB ICFLIB
CALL ICFLIB/RTFMUL
```

- 14** This subroutine is called when the read operation to the program device does not complete. The indication that the timer has ended is checked (RC=0310), and, if it is set, a message displays to the user. The message asks whether to try the read operation again, or to end the job. In this example, the time interval is specified at **10**.

- 15** This subroutine is called for I/O operation errors that are not handled by subroutine **14**. It checks whether the program device is already acquired when an acquire operation is requested, and, if it is, the second acquire is ignored. Otherwise the program ends.

```
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL      10/03/90  14:53:04      Page    1
Compiler . . . . . : IBM AS/400 RPG/400
Command Options:
  Program . . . . . : ICFLIB/RSDMUL
  Source file . . . . : ICFLIB/QICFPUB
  Source member . . . . : *PGM
Text not available for message RXT0073 file QRPGMMSG.
Generation options . . . . . : *NOLIST *NOXREF *NOATR *NODUMP *NOOPTIMIZE
Source listing indentation . . . : *NONE
SAA flagging . . . . . : *NOFLAG
Generation severity level . . . : 9
Print file . . . . . : *LIBL/QSYSPRT
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Text . . . . . : *SRCMBRTXT
Phase trace . . . . . : *NO
Intermediate text dump . . . . . : *NONE
Snap dump . . . . . : *NONE
Codelist . . . . . : *NONE
Ignore decimal data error . . . : *NO
Actual Program Source:
Member . . . . . : RSDMUL
File . . . . . : QICFPUB
Library . . . . . : ICFLIB
Last Change . . . . . : 10/03/90 14:46:07
Description . . . . . : RPG Multi-Session example w/DDS (source)
```

Figure 11-20 (Part 1 of 13). Source Program Example — RSDMUL (User-Defined Formats)

```

5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL          10/03/90 14:53:04          Page      2
SEQUENCE          IND          DO          LAST          PAGE          PROGRAM
NUMBER          *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE          NUM          UPDATE          LINE          ID
          Source Listing
100 H*****
200 H*
300 H* THIS PROGRAM ASSIGNS FOUR SESSIONS AS FOLLOWS:
400 H* 'ICF00' TO INQUIRE ABOUT A CUSTOMER ACCOUNT BEFORE AN
500 H* ORDER IS PROCESSED.
600 H* 'ICF01' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM
700 H* BEING ORDERED (ITEM 000001 THRU 399999).
800 H* 'ICF02' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM
900 H* BEING ORDERED (ITEM 400000 THRU 699999).
1000 H* 'ICF03' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM
1100 H* BEING ORDERED (ITEM 700000 THRU 999999).
1200 H* A DISPLAY DEVICE IS USED TO ENTER THE REQUEST (USING A
1300 H* CUSTOMER AND AN ITEM MENU) THAT IS SENT TO THE REMOTE
1400 H* SYSTEM.
1500 H*
1600 H*****
1700 F*****
1800 F*
1900 F*          FILE SPECIFICATIONS
2000 F*
2100 F* CMNFIL : ICF FILE USED TO SEND A REQUEST TO ONE
2200 F* OF FOUR DIFFERENT TARGET PROGRAMS. MULTIPLE
2300 F* SESSIONS ARE ACTIVE CONCURRENTLY.
2400 F*
2500 F* DSPFIL : DISPLAY FILE USED TO ENTER A REQUEST TO BE
2600 F* SENT TO A REMOTE SYSTEM.
2700 F*
2800 F* THE FOLLOWING INFORMATION IS SPECIFIED AS PART OF THE
2900 F* FILE SPECIFICATION:
3000 F* INFDS : I/O FEEDBACK AREA
3100 F* NUM : SPECIFIES THE MAXIMUM NUMBER OF
3200 F* PROGRAM DEVICES THAT CAN BE ATTACHED
3300 F* TO THIS FILE. A VALUE OF 4 IS
3400 F* SPECIFIED FOR THE ICF FILE.
3500 F* THIS DEFINES THE FILE AS A
3600 F* MULTIPLE DEVICE FILE.
3700 F* ID : 10 CHARACTER PROGRAM DEVICE NAME
3800 F* FIELD WHICH SPECIFIES WHICH PROGRAM
3900 F* DEVICE TO DIRECT THE OPERATION.
4000 F*
4100 F*
4200 F*****
4300 * 1
          H
          *****
4400 FCMNFIL CF E          WORKSTN          10/13/87
4500 F          KINFDS IOFB          10/13/87
4600 F          KINFSR *PSSR          10/14/87
4700 F          KNUM 4          10/13/87
4800 F          KID CMID          10/13/87
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL          10/03/90 14:53:04          Page      3
SEQUENCE          IND          DO          LAST          PAGE          PROGRAM
NUMBER          *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE          NUM          UPDATE          LINE          ID
          RECORD FORMAT(S): LIBRARY ICFLIB FILE CMNFIL.
          EXTERNAL FORMAT ITMRSP RPG NAME ITMRSP
          EXTERNAL FORMAT DTLRSP RPG NAME DTLRSP
          EXTERNAL FORMAT DETACH RPG NAME DETACH
          EXTERNAL FORMAT EOS RPG NAME EOS
          EXTERNAL FORMAT EVKREQ RPG NAME EVKREQ
          EXTERNAL FORMAT ITMREQ RPG NAME ITMREQ
          EXTERNAL FORMAT DTLREQ RPG NAME DTLREQ
4900 FDSPFIL CF E          WORKSTN          10/13/87
5000 F          KINFDS IODS          10/13/87
          RECORD FORMAT(S): LIBRARY ICFLIB FILE DSPFIL.
          EXTERNAL FORMAT CIMENU RPG NAME CIMENU
          EXTERNAL FORMAT DTLMNU RPG NAME DTLMNU
          EXTERNAL FORMAT DTLSCR RPG NAME DTLSCR
          EXTERNAL FORMAT ITMMNU RPG NAME ITMMNU
          EXTERNAL FORMAT ITMSC2 RPG NAME ITMSC2
          EXTERNAL FORMAT ITMSC3 RPG NAME ITMSC3
          EXTERNAL FORMAT TIMOUT RPG NAME TIMOUT
5100 FQPRINT 0 F 132          PRINTER          10/13/87
5200 J*****
5300 J*
5400 J*          INPUT SPECIFICATIONS
5500 J*
5600 J* IODS : REDEFINES THE I/O FEEDBACK AREA OF THE DISPLAY          10/13/87

```

Figure 11-20 (Part 2 of 13). Source Program Example — RSDMUL (User-Defined Formats)

```

5700 I*          FILE. THIS AREA CONTAINS THE NAME OF THE LAST          10/13/87
5800 I*          RECORD PROCESSED. THIS FIELD IS CALLED RECID.          10/13/87
5900 I* IOFB   : REDEFINES THE I/O FEEDBACK AREA FOR THE ICF          10/03/90
6000 I*          FILE.                                                  10/13/87
6100 I*          I*                                                    10/13/87
6200 I*****
6300 I* 2
A000000 INPUT FIELDS FOR RECORD ITMRSP FILE CMNFIL FORMAT ITMRSP.
A000001          1 1 RECITM
A000002          2 70ITEMNO
A000003          8 37 DESC
A000004          38 44QTYLST
A000005          45 51QTYOH
A000006          52 58QTY00
A000007          59 65QTYB0
A000008          66 67 UNITQ
A000009          68 742PR01
A000010          75 810PR05
A000011          82 862UFRT
A000012          87 952SLSTM
A000013          96 1062SLSTY
A000014          107 1152CSTTM
A000015          116 1262CSTTY
A000016          127 1312PR0
A000017          132 1402LOS
A000018          141 196 FILL1
5738R61 V2R1M0 910524          IBM AS/400 RP6/400          ICFLIB/RSDMUL          10/03/90 14:53:04          Page 4
SEQUENCE          IND DO LAST PAGE PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE NUM UPDATE LINE ID
B000000 INPUT FIELDS FOR RECORD DTLRSP FILE CMNFIL FORMAT DTLRSP.
B000001          1 1 RECCUS
B000002          2 70CUSTNO
B000003          8 37 DNAME
B000004          38 430DLSTOR
B000005          44 520DSLSTM
B000006          53 610DSPM01
B000007          62 700DSPM02
B000008          71 790DSPM03
B000009          80 900DSTTYD
B000010          91 930IDEPT
B000011          94 150 FILL2
C000000 INPUT FIELDS FOR RECORD DETACH FILE CMNFIL FORMAT DETACH.
D000000 INPUT FIELDS FOR RECORD EOS FILE CMNFIL FORMAT EOS.
E000000 INPUT FIELDS FOR RECORD EVKREQ FILE CMNFIL FORMAT EVKREQ.
F000000 INPUT FIELDS FOR RECORD ITMREQ FILE CMNFIL FORMAT ITMREQ.
F000001          1 60ITEMNO
G000000 INPUT FIELDS FOR RECORD DTLREQ FILE CMNFIL FORMAT DTLREQ.
G000001          1 60CUSTNO
H000000 INPUT FIELDS FOR RECORD CIMENU FILE DSPFIL FORMAT CIMENU.
H000000 MENU FOR INQUIRY
H000001          3 3 *IN97
H000002          2 2 *IN98
H000003          1 1 *IN99
H000004          4 4 OPTION
I000000 INPUT FIELDS FOR RECORD DTLMNU FILE DSPFIL FORMAT DTLMNU.
I000000 CUSTOMER INQUIRY SCREEN 1
I000001          3 3 *IN97
I000002          2 2 *IN98
I000003          1 1 *IN99
I000004          4 90CUSTNO
J000000 INPUT FIELDS FOR RECORD DTLSCR FILE DSPFIL FORMAT DTLSCR.
J000000 CUSTOMER INQUIRY SCR. #2
J000001          3 3 *IN97
J000002          2 2 *IN98
J000003          1 1 *IN99
K000000 INPUT FIELDS FOR RECORD ITMMNU FILE DSPFIL FORMAT ITMMNU.
K000000 ITEM INQUIRY SCREEN ONE
K000001          3 3 *IN97
K000002          2 2 *IN98
K000003          1 1 *IN99
K000004          4 90ITEMNO
L000000 INPUT FIELDS FOR RECORD ITMSC2 FILE DSPFIL FORMAT ITMSC2.
L000000 ITEM INQUIRY SCREEN TWO
L000001          3 3 *IN97
L000002          2 2 *IN98
L000003          1 1 *IN99
M000000 INPUT FIELDS FOR RECORD ITMSC3 FILE DSPFIL FORMAT ITMSC3.
M000000 ITEM INQUIRY SCREEN 3
M000001          3 3 *IN97
M000002          2 2 *IN98

```

Figure 11-20 (Part 3 of 13). Source Program Example — RSDMUL (User-Defined Formats)


```

M000003          1 1 *IN99
N000000 INPUT FIELDS FOR RECORD TIMEOUT FILE DSPFIL FORMAT TIMEOUT.
N000000 TIME OUT SCREEN
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL          10/03/90 14:53:04          Page 5
SEQUENCE          IND DO          LAST          PAGE          PROGRAM
NUMBER *...1....+...2....+...3....+...4....+...5....+...6....+...7...* USE          NUM          UPDATE          LINE          ID
N000001          3 3 *IN97
N000002          2 2 *IN98
N000003          1 1 *IN99
N000004          4 4 TIMRSP
6400 I10DS          DS          10/13/87
6500 I          1 240 FILL01          10/13/87
6600 I          261 268 RECID          10/13/87
6700 I          271 415 FILL02          10/13/87
6800 I10FB          DS          10/13/87
6900 I          *ROUTINE LOC          10/14/87
7000 I          *STATUS ERR          10/14/87
7100 I          1 240 FILL03          10/13/87
7200 I          38 45 FMTNM          10/13/87
7300 I          273 282 CMID          10/13/87
7400 I          401 404 MAJMIN          10/13/87
7500 I          401 402 MAJCOD          10/13/87
7600 I          403 404 MINCOD          10/13/87
7700 I          261 268 RECID2          10/13/87
7800 I          271 415 FILL04          10/13/87
7900 C*****
8000 C*
8100 C*          CALCULATION SPECIFICATIONS          10/13/87
8200 C*
8300 C* THE DISPLAY PROGRAM DEVICE IS IMPLICITLY ACQUIRED WHEN THE
8400 C* FILE IS OPENED.          10/13/87
8500 C*
8600 C* ALL OF THE ICF PROGRAM DEVICES ARE EXPLICITLY ACQUIRED.          10/03/90
8700 C*
8800 C* EACH OF THE FOUR TARGET PROGRAMS ARE EVOKED TO ESTABLISH
8900 C* TRANSACTIONS WITH THE REMOTE SYSTEMS.          10/13/87
9000 C*
9100 C* THE MAIN INQUIRY MENU (CIMENU) IS WRITTEN TO THE USER'S
9200 C* DISPLAY.          10/13/87
9300 C*
9400 C*****
9500 * 3
9600 C          ENTRY          TAG          10/13/87
9700 C          'ICF00 'ACQ CMNFIL          1ST SESSION          10/13/87
9800 C          'ICF01 'ACQ CMNFIL          2ND SESSION          10/13/87
9900 C          'ICF02 'ACQ CMNFIL          3RD SESSION          10/13/87
10000 C          'ICF03 'ACQ CMNFIL          4TH SESSION          10/13/87
10100 C          MOVEL'ICF00 'CMID          1ST PROGRAM          10/13/87
10200 C          EXSR EVKSR          CALL EVOKE          10/13/87
10300 C          MOVEL'ICF01 'CMID          2ND PROGRAM          10/13/87
10400 C          EXSR EVKSR          CALL EVOKE          10/13/87
10500 C          MOVEL'ICF02 'CMID          3RD PROGRAM          10/13/87
10600 C          EXSR EVKSR          CALL EVOKE          10/13/87
10700 C          MOVEL'ICF03 'CMID          4TH PROGRAM          10/13/87
10800 C          EXSR EVKSR          CALL EVOKE          10/13/87
10900 C          MAIN          TAG          10/13/87
11000 C          WRITECIMENU          10/13/87
11100 C*****
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL          10/03/90 14:53:04          Page 6
SEQUENCE          IND DO          LAST          PAGE          PROGRAM
NUMBER *...1....+...2....+...3....+...4....+...5....+...6....+...7...* USE          NUM          UPDATE          LINE          ID
11200 C*
11300 C*          DETERMINE USER'S REQUEST          10/13/87
11400 C*
11500 C* A READ TO THE DISPLAY DEVICE IS ISSUED TO RECEIVE THE USER'S
11600 C* REQUEST. THE TYPE OF REQUEST MADE IS BASED ON THE DISPLAY
11700 C* FORMAT CURRENTLY ON THE SCREEN. THE RECORD FORMAT NAME IS
11800 C* EXTRACTED FROM THE I/O FEEDBACK AREA AND IS USED TO DETER-
11900 C* MINE WHAT ACTION SHOULD BE TAKEN NEXT.          03/20/89
12000 C*          03/20/89
12100 C*          03/20/89
12200 * 4
12300 C          READRQ          TAG          10/13/87
12400 C          SETOF          8889 TIMEOUT IND          1 2          10/13/87
12500 C          READ DSPFIL          87          3          10/13/87
12600 C          RECID CABEQ'CIMENU 'MENU          MAIN MENU?          03/20/89
12700 C          RECID CABEQ'ITMMNU 'ITMIN          ITEM MENU?          03/20/89
12800 C          RECID CABEQ'ITMSC2 'ITMRTN          ITM SCR?          10/13/87
12900 C          RECID CABEQ'ITMSC3 'ITMRTN          ITM SCR?          10/13/87
13000 C          RECID CABEQ'DTLNU 'DTLIN          DETAIL SCR?          10/13/87

```

Figure 11-20 (Part 4 of 13). Source Program Example — RSDMUL (User-Defined Formats)

```

13100 C          RECID      CABEQ'DTLSCR 'DTLRTN          CUST SCR?          10/13/87
13200 C          WRITECIMENU          MAIN MENU IF          10/13/87
13300 C          GOTO READRQ          THERE IS ERR          10/13/87
13400 C*****
13500 C*
13600 C*          MAIN MENU          10/13/87
13700 C*
13800 C* THE MAIN MENU IS READ TO DETERMINE THE REQUEST ENTERED 10/13/87
13900 C* BY THE USER. IF CMD 1 (*IN99) IS PRESSED, THE PROGRAM 10/13/87
14000 C* IS ENDED. IF OPTION = 1, AN ITEM INQUIRY MENU IS WRITTEN 10/13/87
14100 C* TO THE SCREEN. IF OPTION = 2, A CUSTOMER INQUIRY MENU IS 10/13/87
14200 C* WRITTEN TO THE SCREEN.          10/13/87
14300 C*
14400 C*****
14500 C* 5
14600 C          MENU      TAG          10/13/87
14700 C          *IN99      CABEQ'1'      END          JOB ENDS          10/13/87
14800 C          OPTION      IFEQ '1'          B001 10/13/87
14900 C          WRITEITMNU          ITEM MENU          001 10/13/87
15000 C          ELSE          X001 10/13/87
15100 C          WRITEDTLMNU          CUST MENU          001 10/13/87
15200 C          END          E001 10/13/87
15300 C          GOTO READRQ          10/13/87
15400 C*****
15500 C*
15600 C*          ITEM INQUIRY          10/13/87
15700 C*
15800 C* THE ITEM NUMBER REQUESTED BY THE USER ON THE ITEM INQUIRY 10/13/87
15900 C* SCREEN IS CHECKED. THIS IS DETERMINED BY THE DISPLAY 03/26/89
16000 C* RECORD FORMAT BEING PROCESSED - IN THIS CASE ITMNU. 03/26/89
16100 C*
16200 C* IF CMD 1 (*IN99) IS PRESSED, THE PROGRAM IS ENDED. IF CMD 2 10/13/87
16300 C* IS PRESSED, THE ITEM INQUIRY REQUEST IS CANCELED, AND THE 03/26/89
16400 C* MAIN MENU (CIMENU) IS WRITTEN TO THE SCREEN.          10/13/87
16500 C*          10/13/87
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL          10/03/90 14:53:04          Page 7
SEQUENCE          IND          DO          PAGE          PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE          NUM          UPDATE          LINE          ID
16600 C* IF AN ITEM NUMBER IS ENTERED, A ITEM INQUIRY REQUEST IS 10/13/87
16700 C* SENT TO THE APPROPRIATE REMOTE SYSTEM. THE REMOTE SYSTEM 10/13/87
16800 C* IS SELECTED BASED ON THE ITEM NUMBER REQUESTED.          10/13/87
16900 C*
17000 C* IF AN ERROR OCCURS, THE ERROR IS PRINTED AND THE JOB 10/13/87
17100 C* IS ENDED.          10/13/87
17200 C*
17300 C*****
17400 C* 6
17500 C          ITMIN      TAG          10/13/87
17600 C          *IN99      CABEQ'1'      END          EXIT ON CMD3          10/13/87
17700 C          *IN98      IFEQ '1'          B001 10/13/87
17800 C          WRITECIMENU          MAIN MENU          001 10/13/87
17900 C          GOTO READRQ          001 10/13/87
18000 C          END          E001 10/13/87
18100 C          ITEMNO      CABLE399999          XICF01          10/13/87
18200 C          ITEMNO      CABLE699999          XICF02          10/13/87
18300 C          ITEMNO      CABLE899999          XICF03          10/13/87
18400 C          XICF01      TAG          10/13/87
18500 C          MOVEL'ICF01          'CMID          10/13/87
18600 C          GOTO XITMIN          10/13/87
18700 C          XICF02      TAG          10/13/87
18800 C          MOVEL'ICF02          'CMID          10/13/87
18900 C          GOTO XITMIN          10/13/87
19000 C          XICF03      TAG          10/13/87
19100 C          MOVEL'ICF03          'CMID          10/13/87
19200 C          XITMIN      TAG          10/13/87
19300 C          WRITEITMREQ          INQ W/INVITE          10/13/87
19400 C          MAJCOD      CABGE'04'          ERROR          ERROR RTN          10/13/87
19500 C          TRY89      TAG          10/13/87
19600 C          SETOF          89          3          10/13/87          OQ
19700 C          MOVEL'          'CMID          10/13/87
19800 C          READ CMNFIL          8910RECV ITM INFO          2 3          10/13/87
19900 C          89          EXSR ERRCHK          CHCK ERR INFO          10/13/87
20000 C          MAJMIN      CABGE'0300'          ITMIN          NODATATRYAGN          10/13/87
20100 C          MAJCOD      CABGE'04'          ERROR          ERROR RTN          10/13/87
20200 C          RECID2      CABNE'ITMRSP'          RECERR          PRINT MSG          10/13/87
20300 C*****
20400 C*
20500 C*          PROCESS ITEM INFORMATION          10/13/87
20600 C*
20700 C* THE ITEM RECORD RECEIVED FROM THE TARGET PROGRAM AND THE 10/13/87

```

Figure 11-20 (Part 5 of 13). Source Program Example — RSDMUL (User-Defined Formats)

```

20800 C* INFORMATION ABOUT THE ITEM IS PROCESSED AND DISPLAYED. 10/13/87
20900 C* IF ITEMNO IS 0 OR LESS, IT IS AN INVALID REQUEST AND A FRESH 10/13/87
21000 C* ITEM MENU IS WRITTEN TO THE SCREEN. IF THE REQUEST IS 10/13/87
21100 C* VALID, VALUES ARE CALCULATED BASED ON THE INFORMATION 10/13/87
21200 C* RECEIVED. 10/13/87
21300 C* 10/13/87
21400 C***** 10/13/87
21500 * 7 10/13/87
21600 C ITMOUT TAG 10/13/87
21700 C ITEMNO IFLE 000000 B001 10/13/87
21800 C WRITEITMNU ITEM MENU 001 10/13/87
21900 C GOTO READRQ READ DISPLY 001 10/13/87
5738R61 V2R1M0 910524 IBM AS/400 RP6/400 ICFLIB/RSDMUL 10/03/90 14:53:04 Page 8
SEQUENCE IND DO LAST PAGE PROGRAM
NUMBER *...1...2...3...4...5...6...7...* USE NUM UPDATE LINE ID
22000 C ELSE X001 10/13/87
22100 C Z-ADD0 QAVAIL 70 QTY AVAIL. 001 10/13/87
22200 C ADD QTY0H QAVAIL 001 10/13/87
22300 C SUB QTY00 QAVAIL 001 10/13/87
22400 C ADD QTYB0 QAVAIL 001 10/13/87
22500 C MOVEDESC DSC 001 10/13/87
22600 C MOVE QTY00 QTY0 001 10/13/87
22700 C MOVE QTY0H QTYH 001 10/13/87
22800 C MOVE QTYB0 QTYB 001 10/13/87
22900 C MOVE UNITQ UNT 001 10/13/87
23000 C MOVE PR01 PR1 001 10/13/87
23100 C MOVE PR05 PR5 001 10/13/87
23200 C MOVE UFRT UFR 001 10/13/87
23300 C WRITEITMSC2 DSP DETAIL 001 10/13/87
23400 C GOTO READRQ 001 10/13/87
23500 C***** 10/13/87
23600 C* 10/13/87
23700 C* ADDITIONAL ITEM INFORMATION 10/13/87
23800 C* 10/13/87
23900 C* ADDITIONAL ITEM INFORMATION IS PROCESSED AND THE RESULT 10/13/87
24000 C* DISPLAYED ON THE SCREEN WHEN A RESPONSE IS READ FROM THE 03/20/89
24100 C* DISPLAY WITH AN ITEM SCREEN RECORD FORMAT. 03/20/89
24200 C* 10/13/87
24300 C* IF CMD 1 (*IN99) IS PRESSED, THE PROGRAM IS ENDED. 10/13/87
24400 C* 10/13/87
24500 C* IF CMD 2 (*IN98) IS PRESSED, THE ITEM INQUIRY IS ENDED, AND 03/20/89
24600 C* THE MAIN MENU (CIMENU) IS WRITTEN TO THE SCREEN. 03/20/89
24700 C* 10/13/87
24800 C* IF CMD 3 (*IN97) IS PRESSED, THE ITEM INQUIRY MENU IS 10/14/87
24900 C* WRITTEN ON THE SCREEN. 10/14/87
25000 C* 10/14/87
25100 C* IF 'ENTER' IS PRESSED WHILE SCREEN 2 FOR ITEM REQUESTED IS 10/16/87
25200 C* CURRENTLY DISPLAYED, MORE INFORMATION IS CALCULATED AND 10/16/87
25300 C* DISPLAYED. 10/16/87
25400 C* 10/16/87
25500 C* IF 'ENTER' IS PRESSED WHILE SCREEN 3 FOR ITEM REQUESTED IS 10/16/87
25600 C* CURRENTLY DISPLAYED, THEN THE ITEM INQUIRY MENU IS WRITTEN 03/20/89
25700 C* TO THE SCREEN. 03/20/89
25800 C* 10/13/87
25900 C***** 10/13/87
26000 * 8 10/13/87
26100 C ITMRTN TAG 001 10/13/87
26200 C *IN99 CABEQ'1' END JOB ENDS 001 10/13/87
26300 C *IN98 IFEQ '1' B002 10/13/87
26400 C WRITECIMENU MAIN MENU 002 10/13/87
26500 C GOTO READRQ 002 10/13/87
26600 C END E002 10/13/87
26700 C *IN97 IFEQ '1' CMD 3 ? B002 10/13/87
26800 C RECID IFEQ 'ITMSC2 ' ITM SCR 2 ? B003 10/13/87
26900 C WRITEITMNU YES, THEN ITS 003 10/13/87
27000 C GOTO READRQ ITEM MENU 003 10/13/87
27100 C END E003 10/13/87
27200 C END E002 10/13/87
27300 C RECID IFEQ 'ITMSC3 ' ITM SCR 3 ? B002 10/13/87
5738R61 V2R1M0 910524 IBM AS/400 RP6/400 ICFLIB/RSDMUL 10/03/90 14:53:04 Page 9
SEQUENCE IND DO LAST PAGE PROGRAM
NUMBER *...1...2...3...4...5...6...7...* USE NUM UPDATE LINE ID
27400 C WRITEITMNU YES, THEN ITS 002 10/13/87
27500 C GOTO READRQ ITEM MENU 002 10/13/87
27600 C END E002 10/13/87
27700 C SLSTM SUB CSTM PROFM 92 PROF MONTH 001 10/13/87
27800 C MULT 100 PROFM 001 10/13/87
27900 C SLSTM COMP 0 46 3 001 10/13/87
28000 C N46 PROFM DIV SLSTM PROFM PROF PCT 001 10/13/87
28100 C QTYLST MULT PR01 LOSTS LOST SALES 001 10/13/87

```

Figure 11-20 (Part 6 of 13). Source Program Example — RSDMUL (User-Defined Formats)

```

28200 C          MOVE SLSTM   SLSM          001 10/13/87
28300 C          MOVE SLSTY   SLSY          001 10/13/87
28400 C          MOVE CSTTM   CSTM          001 10/13/87
28500 C          MOVE PROFM   PROFIT        001 10/13/87
28600 C          MOVE CSTTY   CSTY          001 10/13/87
28700 C          WRITEITMSCS          DET ITM INF 001 10/13/87
28800 C          GOTO READRQ          001 10/13/87
28900 C*****
29000 C*
29100 C*          CUSTOMER INQUIRY          10/13/87
29200 C*
29300 C* THE REQUEST FROM THE CUSTOMER INQUIRY MENU IS PROCESSED. 10/13/87
29400 C*
29500 C* IF CMD 1 (*IN99) IS PRESSED, THE PROGRAM IS ENDED. 10/13/87
29600 C*
29700 C* IF CMD 2 (*IN98) IS PRESSED, THE CUSTOMER INQUIRY IS ENDED, 10/13/87
29800 C* AND THE MAIN MENU (CIMENU) IS WRITTEN TO THE SCREEN. 10/13/87
29900 C*
30000 C* IF A CUSTOMER NUMBER IS ENTERED, THE CUSTOMER INQUIRY 10/13/87
30100 C* REQUEST IS SENT TO THE REMOTE SYSTEM. 10/13/87
30200 C*
30300 C* A READ TO THE ICF PROGRAM DEVICE IS ISSUED TO RECEIVE THE 03/20/89
30400 C* INFORMATION FROM THE TARGET PROGRAM. 10/13/87
30500 C*
30600 C* IF AN ERROR OCCURS, THE ERROR IS PRINTED AND THE JOB IS 10/13/87
30700 C* ENDED. 10/13/87
30800 C*
30900 C*****
31000 C* 9
31100 C          DTLIN   TAG          001 10/13/87
31200 C          *IN99  CABEQ'1'   END          JOB ENDS 001 10/13/87
31300 C          *IN98  IFEQ '1'   B002 10/13/87
31400 C          WRITECIMENU          MAIN MENU 002 10/13/87
31500 C          GOTO READRQ          002 10/13/87
31600 C          END          E002 10/13/87
31700 C          EVDTL   TAG          001 10/13/87
31800 C          MOVEV 'ICF00' 'CMID 001 10/13/87
31900 C          WRITEDTLREQ          CUST INQ 001 10/13/87
32000 C          MAJCOD  CABGE'04'   ERROR      ERROR RTN 001 10/13/87
32100 C          TRY88   TAG          001 10/13/87
32200 C          SETOF          88          3 001 10/13/87
32300 C          MOVEV ' ' 'CMID 001 10/13/87
32400 C          READ CMNFIL          8810RCV CUS INF 2 3 001 10/13/87
32500 C          88      EXSR ERRCHK          CHECK ERR 001 10/13/87
32600 C          MAJMIN  CABGE'0300'   EVDTL      NODATATRYAGN 001 10/13/87
32700 C          MAJCOD  CABGE'04'   ERROR      ERROR RTN 001 10/13/87
5738R61 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL 10/03/90 14:53:04 Page 10
SEQUENCE          *...1....+...2....+...3....+...4....+...5....+...6....+...7...* IND DO LAST PAGE PROGRAM
NUMBER          *...1....+...2....+...3....+...4....+...5....+...6....+...7...* USE NUM UPDATE LINE ID
32800 C          RECID2  CABNE'DTLRSP' RECERR          PRINT MSG 001 10/13/87
32900 C*****
33000 C*
33100 C*          PROCESS CUSTOMER INFORMATION          10/13/87
33200 C*
33300 C* THE CUSTOMER DATA RECEIVED FROM THE TARGET PROGRAM IS 03/20/89
33400 C* PROCESSED. IF CUSTOMER NUMBER IS ZERO OR LESS, IT IS AN 03/20/89
33500 C* INVALID REQUEST AND THE MAIN MENU IS WRITTEN TO THE SCREEN. 03/20/89
33600 C* WHEN THE RCVTRNRND INDICATOR(IN90) IS RECEIVED, THE CUSTOMER 03/20/89
33700 C* INFORMATION IS WRITTEN TO THE SCREEN. IF DURING THE READ 03/20/89
33800 C* OPERATION AN ERROR IS RECEIVED, CONTROL GOES TO THE ERROR 03/20/89
33900 C* ROUTINE TO END THE JOB. 03/20/89
34000 C*
34100 C*****
34200 C* 10
34300 C          DTOUT   TAG          001 10/13/87
34400 C          CUSTNO  IFEQ 000000          B002 10/13/87
34500 C          SETOF          66          3 002 10/13/87
34600 C          WRITECIMENU          MAIN MENU 002 10/13/87
34700 C          GOTO READRQ          002 10/13/87
34800 C          END          E002 10/13/87
34900 C          MOVE CUSTNO  CUSTN 001 10/13/87
35000 C          MOVELDNAME  CNAME 001 08/08/89
35100 C          MOVE DLSTOR  DLSTR 001 10/13/87
35200 C          MOVE DSLSTM  DSLSM 001 10/13/87
35300 C          MOVE DSPM01  DSPM1 001 10/13/87
35400 C          MOVE DSPM02  DSPM2 001 10/13/87
35500 C          MOVE DSTTYD  DSTYD 001 10/13/87
35600 C          MOVE IDEPT   DEPT 001 10/13/87
35700 C          WRITEDLSCR          BLD CUS SCR 001 10/13/87
35800 C          GOTO READRQ          001 10/13/87

```

Figure 11-20 (Part 7 of 13). Source Program Example — RSDMUL (User-Defined Formats)

```

35900 C*****
36000 C*
36100 C* THIS ROUTINE HANDLES THE USER'S REQUEST FOLLOWING THE
36200 C* DISPLAY OF THE CUSTOMER INFORMATION. CMD KEY 1 WILL END
36300 C* THE JOB, CMD KEY 2 WILL DISPLAY THE MAIN MENU, AND "ENTER"
36400 C* WILL BRING UP THE CUSTOMER INQUIRY MENU.
36500 C*
36600 C*****
36700 * 11
36800 C DTLR TN TAG 001 10/13/87
36900 C *IN99 CABEQ'1' END JOB ENDS 001 10/13/87
37000 C *IN98 IFEQ '1' B002 10/13/87
37100 C WRITECIMENU MAIN MENU 002 10/13/87
37200 C GOTO READRQ 002 10/13/87
37300 C END E002 10/13/87
37400 C WRITEDLMNU CUSTOMER INQ 001 10/13/87
37500 C GOTO READRQ 001 10/13/87
37600 C*****
37700 C*
37800 C* WHEN AN I/O OPERATION ERROR IS DETECTED, A MESSAGE IS
37900 C* PRINTED AND THE TRANSACTION AND SESSION ARE ENDED FOR EACH
38000 C* OF THE REMOTE SYSTEMS.
38100 C*
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSDMUL 10/03/90 14:53:04 Page 11
SEQUENCE NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE INDO LAST PAGE PROGRAM
38200 C*****
38300 * 12
38400 C RECERR TAG 001 10/13/87
38500 C EXCPTRECER WRONG RECID 001 10/13/87
38600 C GOTO END END PROGRAM 001 10/13/87
38700 C ERROR TAG 001 10/13/87
38800 C EXCPTMERR 001 10/13/87
38900 C END TAG 001 10/13/87
39000 C MOVE'ICF00 'CMID 001 10/13/87
39100 C WRITEDETACH DET 1ST TRN 001 10/13/87
39200 C MOVE'ICF01 'CMID 001 10/13/87
39300 C WRITEDETACH DET 2ND TRN 001 10/13/87
39400 C MOVE'ICF02 'CMID 001 10/13/87
39500 C WRITEDETACH DET 3RD TRN 001 10/13/87
39600 C MOVE'ICF03 'CMID 001 10/13/87
39700 C WRITEDETACH DET 4TH TRN 001 10/13/87
39800 C ABORT TAG 001 10/13/87
39900 C 'ICF00 'REL CMNFIL 86 REL 1ST SES 2 001 10/13/87
40000 C 'ICF01 'REL CMNFIL 86 REL 2ND SES 2 001 10/13/87
40100 C 'ICF02 'REL CMNFIL 86 REL 3RD SES 2 001 10/13/87
40200 C 'ICF03 'REL CMNFIL 86 REL 4TH SES 2 001 10/13/87
40300 C FORCE TAG 001 10/13/87
40400 C SETON LR 1 001 10/13/87
40500 C RETRN 001 10/13/87
40600 C END E001 10/13/87
40700 C*****
40800 C*
40900 C* THIS SUBROUTINE IS CALLED TO EVOKE THE TARGET PROGRAM. THE
41000 C* SAME TARGET PROGRAM (ICFLIB/RTDMULCL) IS EVOKED AT FOUR
41100 C* DIFFERENT REMOTE SYSTEMS. THE PROGRAM DEVICE IDENTIFIES
41200 C* WHICH SESSION SHOULD BE EVOKED. THE PROGRAM DEVICE WAS
41300 C* SPECIFIED IN CMID PRIOR TO CALLING THIS ROUTINE.
41400 C*
41500 C*****
41600 * 13
41700 C EVKSR BEGSR 10/13/87
41800 C MOVE *BLANK PGMID BLANK OUT 10/13/87
41900 C MOVE *BLANK LIB BLANK OUT 10/13/87
42000 C MOVE'RTDMULCL'PGMID PROGR NAME 10/13/87
42100 C MOVE'ICFLIB 'LIB LIBRARY 10/13/87
42200 C WRITEEVKREQ 10/13/87
42300 C MAJCOD CABGE'04' END TO END PGM 10/13/87
42400 C ENSDR 10/13/87
42500 C*****
42600 C*
42700 C* THIS SUBROUTINE IS CALLED TO PERFORM FURTHER CHECKS ON FILE
42800 C* ERRORS RESULTING FROM THE READ OPERATION ISSUED TO THE PRO-
42900 C* GRAM DEVICE. THIS ROUTINE CHECKS FOR THE TIME OUT INDICATION.
43000 C* IF THERE IS A TIME OUT, A MESSAGE IS SENT TO THE USER'S
43100 C* DISPLAY SCREEN REQUESTING ACTION, OTHERWISE PROGRAM ENDS.
43200 C*
43300 C*****
43400 * 14
43500 C ERRCHK BEGSR 10/13/87

```

Figure 11-20 (Part 8 of 13). Source Program Example — RSDMUL (User-Defined Formats)

```

5738R61 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL          10/03/90 14:53:04          Page 12
SEQUENCE
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE
43600 C          MAJMIN  IFEQ '0310'          TIMER EXPD?          B001 10/13/87
43700 C          CHKAGN  TAG                    DISPLAY MSG          001 10/13/87
43800 C          WRITETIMOUT                    86READ REPLY          3 001 10/13/87
43900 C
44000 C 88      TIMRSP  CABEQ'1'          TRY88          CUST INQUIR          001 10/13/87
44100 C 89      TIMRSP  CABEQ'1'          TRY89          ITEM INQUIR          001 10/13/87
44200 C          TIMRSP  IFEQ '2'          END PROGRAM          B002 10/13/87
44300 C          WRITEEOS                    END SESSION          002 10/13/87
44400 C          GOTO FORCE                    END PROGRAM          002 10/13/87
44500 C          END                    E002 10/13/87
44600 C          GOTO CHKAGN                    ASK AGAIN          001 10/13/87
44700 C          END                    E001 10/13/87
44800 C          GOTO ERROR                    ABEND              10/13/87
44900 C          ENDSR                    10/13/87
45000 C*****
45100 C*                    10/14/87
45200 C* THIS IS THE PROGRAM ERROR SUBROUTINE THAT RECEIVES CONTROL          03/20/89
45300 C* WHEN AN ERROR OCCURS AFTER AN I/O OPERATION IS ISSUED TO          03/20/89
45400 C* THE PROGRAM DEVICE AND THERE IS A NON-ZERO VALUE IN THE RPG          03/20/89
45500 C* STATUS FIELD (ERR). THIS ROUTINE CHECKS FOR STATUS VALUES          03/20/89
45600 C* THAT RELATE TO ICF OPERATIONS. IF THE PROGRAM DEVICE          10/03/90
45700 C* IS ALREADY ACQUIRED, THE ERROR IS IGNORED, OTHERWISE, THE          03/20/89
45800 C* PROGRAM IS TERMINATED.          03/20/89
45900 C*                    10/14/87
46000 C*****
46100 * 15                    10/14/87
46200 C          *PSSR  BEGSR                    10/14/87
46300 C          MOVE ' '          RETURN 6          DEFAULT          10/14/87
46400 C          ERR  CABEQ01285          ENDSR          ALREADY ACQ?          10/14/87
46500 C          MOVE '*CANCL'          RETURN          JOB ENDS          10/14/87
46600 C          ENDSR          ENDSRRETURN          BACK TO MAIN          10/14/87
46700 C*****
46800 OQPRINT E 1          MMERR                    02/24/89
46900 0                    21 'COMMUNICATION ERROR.'          02/24/89
47000 0                    34 'MAJOR/MINOR:'          02/24/89
47100 0          MAJCOD          37                    02/24/89
47200 0                    38 '/'          02/24/89
47300 0          MINCOD          40                    02/24/89
47400 0                    49 'FORMAT:'          02/24/89
47500 0          FMTNM          60                    02/24/89
47600 0                    69 'PGMDEV:'          02/24/89
47700 0          CMID          80                    02/24/89
47800 0          E 1          RECER                    02/24/89
47900 0                    20 'UNMATCH RECD FORMAT'          02/24/89
48000 0                    31 '-JOB ENDS.'          02/24/89
48100 0          MAJCOD          37                    02/24/89
48200 0                    38 '/'          02/24/89
48300 0          MINCOD          40                    02/24/89
48400 0                    49 'FORMAT:'          02/24/89
48500 0          RECID2          60                    02/24/89
48600 0                    69 'PGMDEV:'          02/24/89
48700 0          CMID          80                    02/24/89
5738R61 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL          10/03/90 14:53:04          Page 13
* 6103 48701 OVERFLOW INDICATOR 0A ASSIGNED TO FILE QPRINT.
00000000 OUTPUT FIELDS FOR RECORD DETACH FILE CMNFIL FORMAT DETACH.
P0000000 OUTPUT FIELDS FOR RECORD EOS FILE CMNFIL FORMAT EOS.
Q0000000 OUTPUT FIELDS FOR RECORD EVKREQ FILE CMNFIL FORMAT EVKREQ.
Q0000001 PGMID 10 CHAR 10
Q0000002 LIB 20 CHAR 10
R0000000 OUTPUT FIELDS FOR RECORD ITMREQ FILE CMNFIL FORMAT ITMREQ.
R0000001 ITEMNO 6 ZONE 6,0
S0000000 OUTPUT FIELDS FOR RECORD DTLREQ FILE CMNFIL FORMAT DTLREQ.
S0000001 CUSTNO 6 ZONE 6,0
T0000000 OUTPUT FIELDS FOR RECORD CIMENU FILE DSPFIL FORMAT CIMENU.
T0000000 MENU FOR INQUIRY
U0000000 OUTPUT FIELDS FOR RECORD DTLMNU FILE DSPFIL FORMAT DTLMNU.
U0000000 CUSTOMER INQUIRY SCREEN 1
V0000000 OUTPUT FIELDS FOR RECORD DTLSCR FILE DSPFIL FORMAT DTLSCR.
V0000000 CUSTOMER INQUIRY SCR. #2
V0000001 CUSTN 6 CHAR 6
V0000002 DEPT 9 ZONE 3,0
V0000003 DLSTR 15 ZONE 6,0
V0000004 DSLSM 24 ZONE 9,0
V0000005 DSPM1 33 ZONE 9,0
V0000006 DSPM2 42 ZONE 9,0
V0000007 DSPM3 51 ZONE 9,0
V0000008 DSTYD 62 ZONE 11,0

```

Figure 11-20 (Part 9 of 13). Source Program Example — RSDMUL (User-Defined Formats)

```

V000009          CNAME      67 CHAR  5
W000000  OUTPUT FIELDS FOR RECORD ITMMNU FILE DSPFIL FORMAT ITMMNU.
W000000  ITEM INQUIRY SCREEN ONE
X000000  OUTPUT FIELDS FOR RECORD ITMSC2 FILE DSPFIL FORMAT ITMSC2.
X000000  ITEM INQUIRY SCREEN TWO
X000001          DSC       30 CHAR  30
X000002          QAVAIL    37 ZONE  7,0
X000003          QTYH     44 ZONE  7,0
X000004          QTY0     51 ZONE  7,0
X000005          QTYB     58 ZONE  7,0
X000006          UNT      60 CHAR  2
X000007          PR1      67 ZONE  7,2
X000008          PR5      74 ZONE  7,0
X000009          UFR      79 ZONE  5,2
Y000000  OUTPUT FIELDS FOR RECORD ITMSC3 FILE DSPFIL FORMAT ITMSC3.
Y000000  ITEM INQUIRY SCREEN 3
Y000001          SLSM      9 ZONE  9,2
Y000002          SLSY     20 ZONE 11,2
Y000003          CSTM     29 ZONE  9,2
Y000004          CSTY     40 ZONE 11,2
Y000005          PROFIT    45 ZONE  5,2
Y000006          LOSTS    54 ZONE  9,2
Z000000  OUTPUT FIELDS FOR RECORD TIMEOUT FILE DSPFIL FORMAT TIMEOUT.
Z000000  TIME OUT SCREEN
***** END OF SOURCE *****
Additional Diagnostic Messages
* 7089 4400 RPG PROVIDES SEPARATE INDICATOR AREA FOR FILE CMNFIL.
5738R61 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSDMUL 10/03/90 14:53:04 Page 14
Cross Reference
File and Record References:
FILE/RCD DEV/RCD REFERENCES (D=DEFINED)
01 CMNFIL WORKSTN 44000 9700 9800 9900 10000
19800 32400 39900 40000 40100
40200
DETACH 44000 C000000 39100 39300 39500
39700 0000000
DTLREQ 44000 G000000 31900 S000000
DTLRSP 44000 B000000
EOS 44000 D000000 44300 P000000
EVKREQ 44000 E000000 42200 Q000000
ITMREQ 44000 F000000 19300 R000000
ITMRSP 44000 A000000
02 DSPFIL WORKSTN 49000 12500 43900
CIMENU 49000 H000000 11000 13200 17800
26400 31400 34600 37100 T000000
DTLMNU 49000 I000000 15100 37400 U000000
DTLSCR 49000 J000000 35700 V000000
ITMMNU 49000 K000000 14900 21800 26900
27400 W000000
ITMSC2 49000 L000000 23300 X000000
ITMSC3 49000 M000000 28700 Y000000
TIMOUT 49000 N000000 43800 Z000000
03 QPRINT PRINTER 51000 46800 47800 48701
Field References:
FIELD ATTR REFERENCES (M=MODIFIED D=DEFINED)
*IN97 A(1) H000001 I000001 J000001 K000001 L000001
M000001 N000001 26700
*IN98 A(1) H000002 I000002 J000002 K000002 L000002
M000002 N000002 17700 26300 31300
37000
*IN99 A(1) H000003 I000003 J000003 K000003 L000003
M000003 N000003 14700 17600 26200
31200 36900
*PSSR BEGSR 4400 46200D
* 7031 ABORT TAG 39800D
CHKAGN TAG 43700D 44600
CMID A(10) 7300D 10100M 10300M 10500M 10700M
18500M 18800M 19100M 19700M 31800M
32300M 39000M 39200M 39400M 39600M
47700 48700
CNAME A(5) 35000M V000009D
CSTM P(9,2) 28400M Y000003D
CSTTM P(9,2) A000014D 27700 28400
CSTTY P(11,2) A000015D 28600
5738R61 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSDMUL 10/03/90 14:53:04 Page 15
CSTY P(11,2) 28600M Y000004D
CUSTN A(6) 34900M V000001D
CUSTNO P(6,0) B000002D G000001D I000004D 34400 34900
S000001D
DEPT P(3,0) 35600M V000002D

```

Figure 11-20 (Part 10 of 13). Source Program Example — RSDMUL (User-Defined Formats)

	DESC	A(30)	A000003D	22500						
	DLSTOR	P(6,0)	B000004D	35100						
	DLSTR	P(6,0)	35100M	V000003D						
	DNAME	A(30)	B000003D	35000						
	DSC	A(30)	22500M	X000001D						
	DSLSM	P(9,0)	35200M	V000004D						
	DSLSTM	P(9,0)	B000005D	35200						
	DSPM01	P(9,0)	B000006D	35300						
	DSPM02	P(9,0)	B000007D	35400						
* 7031	DSPM03	P(9,0)	B000008D							
	DSPM1	P(9,0)	35300M	V000005D						
	DSPM2	P(9,0)	35400M	V000006D						
	DSPM3	P(9,0)	V000007D							
	DSTTYD	P(11,0)	B000009D	35500						
	DSTYD	P(11,0)	35500M	V000008D						
	DTLIN	TAG	13000	31100D						
	DTLRTN	TAG	13100	36800D						
* 7031	DTOUT	TAG	34300D							
	END	TAG	14700	17600	26200	31200	36900			
			38600	38900D	42300					
* 7031	ENDPSR	ENDSR	46400	46600D						
	ENTRY	TAG	9600D							
	ERR	Z(5,0)	7000D	46400						
	ERRCHK	BEGSR	19900	32500	43500D					
	ERROR	TAG	19400	20100	32000	32700	38700D			
			44800							
	EVDTL	TAG	31700D	32600						
	EVKSR	BEGSR	10200	10400	10600	10800	41700D			
* 7031	FILL01	A(240)	6500D							
* 7031	FILL02	A(145)	6700D							
* 7031	FILL03	A(240)	7100D							
* 7031	FILL04	A(145)	7800D							
* 7031	FILL1	A(56)	A000018D							
* 7031	FILL2	A(57)	B000011D							
	FMTNM	A(8)	7200D	47500						
	FORCE	TAG	40300D	44400						
	IDEPT	P(3,0)	B000010D	35600						
	IODS	DS(415)	4900	6400D						
	IOFB	DS(415)	4400	6800D						
	ITEMNO	P(6,0)	A000002D	F000001D	K000004D	18100	18200			
			18300	21700	R000001D					
* 7031	ITMIN	TAG	12700	17500D	20000					
	ITMOUT	TAG	21600D							
	ITMRTN	TAG	12800	12900	26100D					
	LIB	A(10)	41900M	42100M	Q000002D					
* 7031	LOC	A(8)	6900D							
* 7031	LOS	P(9,2)	A000017D							
	LOSTS	P(9,2)	28100M	Y000006D						
* 7031	MAIN	TAG	10900D							
	MAJCOD	A(2)	7500D	19400	20100	32000	32700			
			42300	47100	48100					
5738RG1	V2R1M0	910524	IBM AS/400	RP6/400				ICFLIB/RSDMUL	10/03/90	14:53:04
	MAJMIN	A(4)	7400D	20000	32600	43600				
	MENU	TAG	12600	14600D						
	MINCOD	A(2)	7600D	47300	48300					
	MMERR	EXCPT	38800	46800						
	OPTION	A(1)	H000004D	14800						
* 7031	PGMID	A(10)	41800M	42000M	Q000001D					
	PRO	P(5,2)	A000016D							
	PROFIT	P(5,2)	28500M	Y000005D						
	PROFM	P(9,2)	27700D	27800M	28000	28000M	28500			
	PR01	P(7,2)	A000009D	23000	28100					
	PR05	P(7,0)	A000010D	23100						
	PR1	P(7,2)	23000M	X000007D						
	PR5	P(7,0)	23100M	X000008D						
	QAVAIL	P(7,0)	22100D	22200M	22300M	22400M	X000002D			
	QTYB	P(7,0)	22800M	X000005D						
	QTYB0	P(7,0)	A000007D	22400	22800					
	QTYH	P(7,0)	22700M	X000003D						
	QTYLST	P(7,0)	A000004D	28100						
	QTYO	P(7,0)	22600M	X000004D						
	QTYOH	P(7,0)	A000005D	22200	22700					
	QTYO0	P(7,0)	A000006D	22300	22600					
	READRQ	TAG	12300D	13300	15300	17900	21900			
			23400	26500	27000	27500	28800			
			31500	34700	35800	37200	37500			
* 7031	RECCUS	A(1)	B000001D							
	RECER	EXCPT	38500	47800						
	RECERR	TAG	20200	32800	38400D					

Figure 11-20 (Part 11 of 13). Source Program Example — RSDMUL (User-Defined Formats)


```

RECID      A(8)      6600D 12600 12700 12800 12900
              13000 13100 26800 27300
* 7031 RECID2     A(8)      7700D 20200 32800 48500
RECITM     A(1)     A000001D
RETURN     A(6)     46300D 46500M 46600
SLSM       P(9,2)   28200M Y000001D
SLSTM      P(9,2)   A000012D 27700 27900 28000 28200
SLSTY      P(11,2) A000013D 28300
SLSY       P(11,2) 28300M Y000002D
TIMRSP     A(1)     N000004D 44000 44100 44200
TRY88      TAG      32100D 44000
TRY89      TAG      19500D 44100
UFR        P(5,2)   23200M X000009D
UFRT       P(5,2)   A000011D 23200
UNITQ      A(2)     A000008D 22900
UNT        A(2)     22900M X000006D
XICF01     TAG      18100 18400D
XICF02     TAG      18200 18700D
XICF03     TAG      18300 19000D
XITMIN     TAG      18600 18900 19200D
*BLANK     LITERAL 41800 41900
'         ' LITERAL 19700 32300
'         ' LITERAL 46300
'*CANCL'   LITERAL 46500
'CIMENU'   LITERAL 12600
'DTLMNU'   LITERAL 13000
'DTLRSP'   LITERAL 32800
'DTLSCR'   LITERAL 13100
'ICFLIB'   LITERAL 42100
'ICF00'    LITERAL 9700 10100 31800 39000 39900
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSDMUL 10/03/90 14:53:04 Page 17
'ICF01'    LITERAL 9800 10300 18500 39200 40000
'ICF02'    LITERAL 9900 10500 18800 39400 40100
'ICF03'    LITERAL 10000 10700 19100 39600 40200
'ITMMNU'   LITERAL 12700
'ITMRSP'   LITERAL 20200
'ITMSC2'   LITERAL 12800 26800
'ITMSC3'   LITERAL 12900 27300
'RTDMULCL' LITERAL 42000
'0300'    LITERAL 20000 32600
'0310'    LITERAL 43600
'04'      LITERAL 19400 20100 32000 32700 42300
'1'       LITERAL 14700 14800 17600 17700 26200
              26300 26700 31200 31300 36900
              37000 44000 44100
'2'       LITERAL 44200
0         LITERAL 22100 27900
000000    LITERAL 21700 34400
01285     LITERAL 46400
100       LITERAL 27800
3999999   LITERAL 18100
6999999   LITERAL 18200
8999999   LITERAL 18300
Indicator References:
INDICATOR REFERENCES (M=MODIFIED D=DEFINED)
*IN        H000001 H000002 H000003 I000001 I000002 I000003
           J000001 J000002 J000003 K000001 K000002 K000003
           L000001 L000002 L000003 M000001 M000002 M000003
           N000001 N000002 N000003 14700 17600 17700
           26200 26300 26700 31200 31300 36900
           37000
LR         40400M
0A         5100D 48701
* 7031 10 19800M 32400M
46         27900M 28000
* 7031 66 34500M
* 7031 86 39900M 40000M 40100M 40200M 43900M
* 7031 87 12500M
88         12400M 32200M 32400M 32500 44000
89         12400M 19600M 19800M 19900 44100
* 7031 90
97         H000001 I000001 J000001 K000001 L000001 M000001
           N000001 26700
98         H000002 I000002 J000002 K000002 L000002 M000002
           N000002 17700 26300 31300 37000
99         H000003 I000003 J000003 K000003 L000003 M000003
           N000003 14700 17600 26200 31200 36900
***** END OF CROSS REFERENCE *****

```

Figure 11-20 (Part 12 of 13). Source Program Example — RSDMUL (User-Defined Formats)

```

5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL          10/03/90 14:53:04          Page    18
                                M e s s a g e   S u m m a r y
* QRG6103 Severity: 00      Number: 1
  Message . . . . : No Overflow Indicator is specified but an
                    indicator is assigned to a file and automatic skip to 6 is
                    generated.
* QRG7031 Severity: 00      Number: 22
  Message . . . . : The Name or indicator is not referenced.
* QRG7089 Severity: 00      Number: 1
  Message . . . . : The RPG provides Separate-Indicator area for
                    file.
      * * * * * E N D   O F   M E S S A G E   S U M M A R Y   * * * * *
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSDMUL          10/03/90 14:53:04          Page    19
                                F i n a l   S u m m a r y
Message Count: (by Severity Number)
      TOTAL    00    10    20    30    40    50
      24     24     0     0     0     0     0
Program Source Totals:
  Records . . . . . : 487
  Specifications . . . . . : 245
  Table Records . . . . . : 0
  Comments . . . . . : 242
PRM has been called.
Program RSDMUL is placed in library ICFLIB. 00 highest Error-Severity-Code.
      * * * * * E N D   O F   C O M P I L A T I O N   * * * * *

```

Figure 11-20 (Part 13 of 13). Source Program Example — RSDMUL (User-Defined Formats)

```

5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSFMUL          10/03/90 14:55:01      Page      1
Compiler . . . . . : IBM AS/400 RPG/400
Command Options:
  Program . . . . . : ICFLIB/RSFMUL
  Source file . . . . : ICFLIB/QICFPUB
  Source member . . . . : *PGM
Text not available for message RXT0073 file QRPMSG6.
Generation options . . . . . : *NOLIST *NOXREF *NOATR *NODUMP *NOOPTIMIZE
Source listing indentation . . . : *NONE
SAA flagging . . . . . : *NOFLAG
Generation severity level . . . . : 9
Print file . . . . . : *LIBL/QSYSPRT
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Text . . . . . : *SRCMBRTXT
Phase trace . . . . . : *NO
Intermediate text dump . . . . . : *NONE
Snap dump . . . . . : *NONE
Codelist . . . . . : *NONE
Ignore decimal data error . . . . : *NO
Actual Program Source:
Member . . . . . : RSFMUL
File . . . . . : QICFPUB
Library . . . . . : ICFLIB
Last Change . . . . . : 10/03/90 14:47:52
Description . . . . . : RPG Multi-Session example w/FORMAT (source)
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSFMUL          10/03/90 14:55:01      Page      2
SEQUENCE          INO          DO          LAST          PAGE          PROGRAM
NUMBER          *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE          NUM          UPDATE          LINE          ID
Source Listing
100 H*****
200 H*
300 H* THIS PROGRAM ASSIGNS FOUR SESSIONS AS FOLLOWS:
400 H* 'ICF00' TO INQUIRE ABOUT A CUSTOMER ACCOUNT BEFORE AN
500 H* ORDER IS PROCESSED.
600 H* 'ICF01' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM
700 H* BEING ORDERED (ITEM 000001 THRU 399999).
800 H* 'ICF02' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM
900 H* BEING ORDERED (ITEM 400000 THRU 699999).
1000 H* 'ICF03' TO INQUIRE ABOUT THE INVENTORY STATUS OF AN ITEM
1100 H* BEING ORDERED (ITEM 700000 THRU 999999).
1200 H* A DISPLAY DEVICE IS USED TO ENTER THE REQUEST ( USING A
1300 H* CUSTOMER AND AN ITEM MENU) THAT IS SENT TO THE REMOTE
1400 H* SYSTEM.
1500 H*
1600 F*****
1700 F*
1800 F* FILE SPECIFICATIONS
1900 F*
2000 F* CMNFIL : ICF FILE USED TO SEND A REQUEST TO ONE
2100 F* OF FOUR DIFFERENT TARGET PROGRAMS. MULTIPLE
2200 F* SESSIONS ARE ACTIVE CONCURRENTLY.
2300 F*
2400 F* DSPFIL : DISPLAY FILE USED TO ENTER A REQUEST TO BE
2500 F* SENT TO A REMOTE SYSTEM.
2600 F*
2700 F* THE FOLLOWING INFORMATION IS SPECIFIED AS PART OF THE
2800 F* FILE SPECIFICATION:
2900 F* INFDS : I/O FEEDBACK AREA
3000 F* NUM : SPECIFIES THE MAXIMUM NUMBER OF PRO-
3100 F* GRAM DEVICES THAT CAN BE ATTACHED TO
3200 F* THIS FILE. A VALUE OF 4 IS SPECIFIED
3300 F* FOR THE ICF FILE. THIS DEFINES
3400 F* THE FILE AS A MULTIPLE DEVICE FILE.
3500 F* ID : 10 CHARACTER PROGRAM DEVICE NAME FIELD
3600 F* WHICH SPECIFIES WHICH PROGRAM DEVICE TO
3700 F* DIRECT THE OPERATION.
3800 F* ID : 10 CHARACTER PROGRAM DEVICE NAME
3900 F* FIELD THAT SPECIFIES WHICH PROGRAM
4000 F* DEVICE TO DIRECT THE OPERATION.
4100 F*
4200 F*
4300 F*****
4400 * 1
H
4500 FCMNFIL CF F 150 WORKSIN
4600 F KINFDS IOFB
4700 F KINFSR *PSSR

```

Figure 11-21 (Part 1 of 12). Source Program Example — RSFMUL (System-Supplied Formats)

```

4800 F                                KNUM      4                                10/13/87
5738RG1 V2R1M0 910524                IBM AS/400 RPG/400                ICFLIB/RSFMUL 10/03/90 14:55:01 Page 3
SEQUENCE                               IND      DO      LAST      PAGE      PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE  NUM  UPDATE  LINE  ID
4900 F                                KID      CMID                                10/13/87
5000 F DSPFIL CF E                    WORKSTN                                     10/13/87
5100 F                                KINFDS  IODS                                10/13/87
RECORD FORMAT(S): LIBRARY ICFLIB FILE DSPFIL.
EXTERNAL FORMAT CIMENU RPG NAME CIMENU
EXTERNAL FORMAT DTLMNU RPG NAME DTLMNU
EXTERNAL FORMAT DTLSCR RPG NAME DTLSCR
EXTERNAL FORMAT ITMMNU RPG NAME ITMMNU
EXTERNAL FORMAT ITMSC2 RPG NAME ITMSC2
EXTERNAL FORMAT ITMSC3 RPG NAME ITMSC3
EXTERNAL FORMAT TIMOUT RPG NAME TIMOUT
5200 FQPRINT 0 F 132                  PRINTER                                     10/13/87
5300 I*****
5400 I*
5500 I*          INPUT SPECIFICATIONS
5600 I*
5700 I* IODS : REDEFINES THE I/O FEEDBACK AREA OF THE DISPLAY
5800 I*          FILE. THIS AREA CONTAINS THE NAME OF THE LAST
5900 I*          RECORD PROCESSED. THIS FIELD IS CALLED RECID.
6000 I* IOFB : REDEFINES THE I/O FEEDBACK AREA FOR THE ICF
6100 I*          FILE.
6200 I*
6300 I*****
6400 I* 2
6500 ICMNFIL NS 80 1 CC
6600 I          1 1 RECCUS
6700 I          2 70CUSTNO
6800 I          8 37 DNAME
6900 I          38 43 DLSTOR
7000 I          44 52 DSLSTM
7100 I          53 61 DSPM01
7200 I          62 70 DSPM02
7300 I          71 79 DSPM03
7400 I          80 90 DSTTYD
7500 I          91 93 IDEPT
7600 I          94 150 FILL20
7700 I          NS 81 1 CI
7800 I          1 1 RECITM
7900 I          2 70ITEMNO
8000 I          8 37 DESC
8100 I          38 440QTYLST
8200 I          45 510QTYOH
8300 I          52 580QTY00
8400 I          59 650QTYB0
8500 I          66 67 UNIT
8600 I          68 742PR01
8700 I          75 800PR05
8800 I          81 852UFRT
8900 I          86 942SLSTM
9000 I          95 1052SLSTY
9100 I          106 1142CSTTM
5738RG1 V2R1M0 910524                IBM AS/400 RPG/400                ICFLIB/RSFMUL 10/03/90 14:55:01 Page 4
SEQUENCE                               IND      DO      LAST      PAGE      PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE  NUM  UPDATE  LINE  ID
9200 I          115 1252CSTTY
9300 I          126 1302PRO
9400 I          131 1392LOS
9500 I          140 150 FILL10
9600 I*****
A000000 INPUT FIELDS FOR RECORD CIMENU FILE DSPFIL FORMAT CIMENU.
A000000 MENU FOR INQUIRY
A000001          3 3 *IN97
A000002          2 2 *IN98
A000003          1 1 *IN99
A000004          4 4 OPTION
B000000 INPUT FIELDS FOR RECORD DTLMNU FILE DSPFIL FORMAT DTLMNU.
B000000 CUSTOMER INQUIRY SCREEN 1
B000001          3 3 *IN97
B000002          2 2 *IN98
B000003          1 1 *IN99
B000004          4 90CUSTNO
C000000 INPUT FIELDS FOR RECORD DTLSCR FILE DSPFIL FORMAT DTLSCR.
C000000 CUSTOMER INQUIRY SCR. #2
C000001          3 3 *IN97
C000002          2 2 *IN98
C000003          1 1 *IN99

```

Figure 11-21 (Part 2 of 12). Source Program Example — RSFMUL (System-Supplied Formats)

```

D000000 INPUT FIELDS FOR RECORD ITMMNU FILE DSPFIL FORMAT ITMMNU.
D000000 ITEM INQUIRY SCREEN ONE
D000001          3 3 *IN97
D000002          2 2 *IN98
D000003          1 1 *IN99
D000004          4 90ITEMNO
E000000 INPUT FIELDS FOR RECORD ITMSC2 FILE DSPFIL FORMAT ITMSC2.
E000000 ITEM INQUIRY SCREEN TWO
E000001          3 3 *IN97
E000002          2 2 *IN98
E000003          1 1 *IN99
F000000 INPUT FIELDS FOR RECORD ITMSC3 FILE DSPFIL FORMAT ITMSC3.
F000000 ITEM INQUIRY SCREEN 3
F000001          3 3 *IN97
F000002          2 2 *IN98
F000003          1 1 *IN99
G000000 INPUT FIELDS FOR RECORD TIMOUT FILE DSPFIL FORMAT TIMOUT.
G000000 TIME OUT SCREEN
G000001          3 3 *IN97
G000002          2 2 *IN98
G000003          1 1 *IN99
G000004          4 4 TIMRSP
9700 I10DS      DS
9800 I          1 240 FILL01
9900 I          261 268 RECID
10000 I         271 415 FILL02
10100 I10FB     DS
10200 I          *ROUTINE LOC
10300 I          *STATUS ERR
10400 I          1 240 FILL03
10500 I          38 45 FMTNM
10600 I         273 282 CMID
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSFMUL          10/03/90 14:55:01          Page
SEQUENCE          IND DO LAST PAGE PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE NUM UPDATE LINE ID
10700 I          401 404 MAJMIN          10/13/87
10800 I          401 402 MAJCOD          10/13/87
10900 I          403 404 MINCOD          10/13/87
11000 I          261 268 RECID2          10/13/87
11100 I          271 415 FILL04          10/13/87
11200 C*****
11300 C*
11400 C*          C A L C U L A T I O N   S P E C I F I C A T I O N S
11500 C*          T H E   D I S P L A Y   D E V I C E   I S   I M P L I C I T L Y   A C Q U I R E D   W H E N   T H E
11600 C*          F I L E   I S   O P E N E D .
11700 C*
11800 C*          A L L   O F   T H E   P R O G R A M   D E V I C E S   A R E   E X P L I C I T L Y   A C Q U I R E D .
11900 C*
12000 C*          E A C H   O F   T H E   F O U R   T A R G E T   P R O G R A M S   A R E   E V O K E D   T O   E S T A B L I S H
12100 C*          T R A N S A C T I O N S   W I T H   T H E   R E M O T E   S Y S T E M S .
12200 C*
12300 C*          T H E   M A I N   I N Q U I R Y   M E N U   ( C I M E N U )   I S   W R I T T E N   T O   T H E   U S E R ' S
12400 C*          D I S P L A Y .
12500 C*
12600 C*****
12700 * 3
12800 C          B E G I N          T A G
12900 C          ' I C F 0 0   ' A C Q   C M N F I L          1 S T   S E S S I O N
13000 C          ' I C F 0 1   ' A C Q   C M N F I L          2 N D   S E S S I O N
13100 C          ' I C F 0 2   ' A C Q   C M N F I L          3 R D   S E S S I O N
13200 C          ' I C F 0 3   ' A C Q   C M N F I L          4 T H   S E S S I O N
13300 C          M O V E L ' I C F 0 0   ' C M I D          1 S T   P R O G R A M
13400 C          E X S R   E V K S R          C A L L   E V O K E
13500 C          M O V E L ' I C F 0 1   ' C M I D          2 N D   P R O G R A M
13600 C          E X S R   E V K S R          C A L L   E V O K E
13700 C          M O V E L ' I C F 0 2   ' C M I D          3 R D   P R O G R A M
13800 C          E X S R   E V K S R          C A L L   E V O K E
13900 C          M O V E L ' I C F 0 3   ' C M I D          4 T H   P R O G R A M
14000 C          E X S R   E V K S R          C A L L   E V O K E
14100 C          M A I N          T A G
14200 C          W R I T E C I M E N U
14300 C*****
14400 C*
14500 C*          D E T E R M I N E   U S E R ' S   R E Q U E S T
14600 C*
14700 C*          A   R E A D   O P E R A T I O N   I S   I S S U E D   T O   T H E   D I S P L A Y   P R O G R A M   D E V I C E
14800 C*          T O   R E C E I V E   T H E   U S E R ' S   R E Q U E S T .   T H E   T Y P E   O F   R E Q U E S T   M A D E   I S
14900 C*          B A S E D   O N   T H E   D I S P L A Y   F O R M A T   C U R R E N T L Y   O N   T H E   S C R E E N .   T H E
15000 C*          R E C O R D   F O R M A T   N A M E   I S   E X T R A C T E D   F R O M   T H E   I / O   F E E D B A C K   A R E A
15100 C*          A N D   U S E D   T O   D E T E R M I N E   W H A T   A C T I O N   S H O U L D   B E   T A K E N   N E X T .

```

Figure 11-21 (Part 3 of 12). Source Program Example — RSFMUL (System-Supplied Formats)

```

15200 C* 10/13/87
15300 C***** 10/13/87
15400 * 4 10/13/87
15500 C READRQ TAG 10/13/87
15600 C READ DSPFIL 88 3 10/13/87
15700 C RECID CABEQ'CIMENU 'MENU MAIN MENU 10/13/87
15800 C RECID CABEQ'ITMMNU 'ITMIN AT ITEM SCR? 10/13/87
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSFMUL 10/03/90 14:55:01 Page 6
SEQUENCE NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* IND DO LAST PAGE PROGRAM
UPDATE LINE ID
15900 C RECID CABEQ'ITMSC2 'ITMRTN AT ITM SCR? 10/13/87
16000 C RECID CABEQ'ITMSC3 'ITMRTN AT ITM SCR? 10/13/87
16100 C RECID CABEQ'DTLMNU 'DTLIN FOR DETAIL? 10/13/87
16200 C RECID CABEQ'DTLSCR 'DTLRTN CUST SCR? 10/13/87
16300 C WRITECIMENU MAIN MENU 10/13/87
16400 C GOTO READRQ THERE IS ERR 10/13/87
16500 C***** 10/13/87
16600 C* 10/13/87
16700 C* MAIN MENU 10/13/87
16800 C* 10/13/87
16900 C* THE MAIN MENU IS READ TO DETERMINE THE REQUEST ENTERED BY 03/20/89
17000 C* THE USER. IF CMD 1 (*IN99) IS PRESSED, THE PROGRAM IS ENDED. 03/20/89
17100 C* IF OPTION = 1, AN ITEM INQUIRY MENU IS WRITTEN TO THE SCREEN. 03/20/89
17200 C* IF OPTION = 2, A CUSTOMER INQUIRY MENU IS WRITTEN TO THE 03/20/89
17300 C* SCREEN. 03/20/89
17400 C* 10/13/87
17500 C***** 10/13/87
17600 * 5 10/13/87
17700 C MENU TAG 10/13/87
17800 C *IN99 CABEQ'1' END END PROGRAM 10/13/87
17900 C OPTION IFEQ '1' B001 10/13/87
18000 C WRITEITMMNU ITEM MENU 001 10/13/87
18100 C ELSE X001 10/13/87
18200 C WRITEDTLMNU CUST MENU 001 10/13/87
18300 C END E001 10/13/87
18400 C GOTO READRQ 10/13/87
18500 C***** 10/13/87
18600 C* 10/13/87
18700 C* ITEM INQUIRY 10/13/87
18800 C* 10/13/87
18900 C* THE ITEM NUMBER REQUESTED BY THE USER ON THE ITEM INQUIRY 10/13/87
19000 C* SCREEN IS CHECKED. THIS IS DETERMINED BY THE DISPLAY 03/20/89
19100 C* RECORD FORMAT BEING PROCESSED - IN THIS CASE ITMMNU. 03/20/89
19200 C* 10/13/87
19300 C* IF CMD 1 (*IN99) IS PRESSED, THE PROGRAM IS ENDED. IF CMD 2 10/13/87
19400 C* IS PRESSED, THE ITEM INQUIRY REQUEST IS CANCELED, AND THE 10/13/87
19500 C* MAIN MENU (CIMENU) IS WRITTEN TO THE SCREEN. 10/13/87
19600 C* 10/13/87
19700 C* IF AN ITEM NUMBER IS ENTERED, A ITEM INQUIRY REQUEST IS 10/13/87
19800 C* SENT TO THE APPROPRIATE REMOTE SYSTEM. THE REMOTE SYSTEM 10/13/87
19900 C* IS SELECTED BASED ON THE ITEM NUMBER REQUESTED. 10/13/87
20000 C* 10/13/87
20100 C* IF AN ERROR OCCURS, THE ERROR IS PRINTED AND THE JOB 10/13/87
20200 C* IS ENDED. 10/13/87
20300 C* 10/13/87
20400 C***** 10/13/87
20500 * 6 10/13/87
20600 C ITMIN TAG 10/13/87
20700 C *IN99 CABEQ'1' END EXIT ON CMD3 10/13/87
20800 C *IN98 IFEQ '1' B001 10/13/87
20900 C WRITECIMENU MAIN MENU 001 10/13/87
21000 C GOTO READRQ 001 10/13/87
21100 C END E001 10/13/87
21200 C ITEMNO CABLE399999 XICF01 10/13/87
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSFMUL 10/03/90 14:55:01 Page 7
SEQUENCE NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* IND DO LAST PAGE PROGRAM
UPDATE LINE ID
21300 C ITEMNO CABLE699999 XICF02 10/13/87
21400 C ITEMNO CABLE899999 XICF03 10/13/87
21500 C XICF01 TAG 10/13/87
21600 C MOVEV'ICF01 'CMID 10/13/87
21700 C GOTO XITMIN 10/13/87
21800 C XICF02 TAG 10/13/87
21900 C MOVEV'ICF02 'CMID 10/13/87
22000 C GOTO XITMIN 10/13/87
22100 C XICF03 TAG 10/13/87
22200 C MOVEV'ICF03 'CMID 10/13/87
22300 C XITMIN TAG 10/13/87
22400 C EXCPTITEMRQ INQ W/INV 10/13/87
22500 C MAJCOD CABGE'04' ERROR ERROR RTN 10/13/87

```

Figure 11-21 (Part 4 of 12). Source Program Example — RSFMUL (System-Supplied Formats)

```

22600 C          TRY89  TAG                10/13/87
22700 C          SETOF                89          1          10/13/87
22800 C          MOVEL ' '          RECITM          RESET RECID          10/13/87
22900 C          MOVEL '          'CMID          10/13/87
23000 C          READ CMNFIL          8910          2 3          10/13/87
23100 C          89          EXSR ERRCHK          FILE ERROR?          10/13/87
23200 C          MAJMIN          CABGE '0300'          ITMIN          NODATA?          10/13/87
23300 C          N81          EXCPTNOTITM          REC NOT FD?          10/13/87
23400 C          SETOF                81          1          10/13/87
23500 C          MAJCOD          CABGE '04'          ERROR          ERROR RTN          10/13/87
23600 C          RECITM          CABNE 'I'          RECERR          PRINT MSG          10/13/87
23700 C*****
23800 C*
23900 C*          PROCESS ITEM INFORMATION          10/13/87
24000 C*
24100 C*          THE ITEM RECORD RECEIVED FROM THE TARGET PROGRAM AND THE          10/13/87
24200 C*          INFORMATION ABOUT THE ITEM IS PROCESSED AND DISPLAYED. IF          03/20/89
24300 C*          ITEMNO IS 0 OR LESS, IT IS AN INVALID REQUEST AND A FRESH          03/20/89
24400 C*          ITEM MENU IS WRITTEN TO THE SCREEN. IF REQUEST IS VALID,          03/20/89
24500 C*          VALUES ARE CALCULATED BASED ON THE INFORMATION RECEIVED.          03/20/89
24600 C*
24700 C*****
24800 C* 7
24900 C          ITMOUT          TAG                10/13/87
25000 C          ITEMNO          IFLE 000000          B001          10/13/87
25100 C          WRITEITMMNU          ITEM MENU          001          10/13/87
25200 C          GOTO READRQ          READ DISPLAY          001          10/13/87
25300 C          ELSE          X001          10/13/87
25400 C          Z-ADD0          QAVAIL          70          QTY AVAIL.          001          10/13/87
25500 C          ADD QTY0H          QAVAIL          001          10/13/87
25600 C          SUB QTY00          QAVAIL          001          10/13/87
25700 C          ADD QTYB0          QAVAIL          001          10/13/87
25800 C          MOVEDESC          DSC                001          10/13/87
25900 C          MOVE QTY00          QTY0          001          10/13/87
26000 C          MOVE QTY0H          QTYH          001          10/13/87
26100 C          MOVE QTYB0          QTYB          001          10/13/87
26200 C          MOVE UNIT          UNT                001          10/13/87
26300 C          MOVE PR01          PR1                001          10/13/87
26400 C          MOVE PR05          PR5                001          10/13/87
26500 C          MOVE UFRT          UFR                001          10/13/87
26600 C          WRITEITMSC2          ITEM DETAIL          001          10/13/87
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RSFMUL          10/03/90          14:55:01          Page          8
SEQUENCE          *...1....+....2....+....3....+....4....+....5....+....6....+....7...*          IND          DO          LAST          PAGE          PROGRAM
NUMBER          *...1....+....2....+....3....+....4....+....5....+....6....+....7...*          USE          NUM          UPDATE          LINE          ID
26700 C          GOTO READRQ          001          10/13/87
26800 C          END          E001          10/13/87
26900 C*****
27000 C*
27100 C*          ADDITIONAL ITEM INFORMATION          10/13/87
27200 C*
27300 C*          ADDITIONAL ITEM INFORMATION IS PROCESSED AND THE RESULT          10/13/87
27400 C*          IS DISPLAYED ON THE SCREEN WHEN A RESPONSE IS READ          03/20/89
27500 C*          FROM THE DISPLAY WITH AN ITEM SCREEN RECORD FORMAT.          03/20/89
27600 C*
27700 C*          IF CMD 1 (*IN99) IS PRESSED, THE PROGRAM IS ENDED.          10/13/87
27800 C*
27900 C*          IF CMD 2 (*IN98) IS PRESSED, THE ITEM INQUIRY IS ENDED,          03/20/89
28000 C*          AND THE MAIN MENU (CIMENU) IS WRITTEN TO THE SCREEN.          03/20/89
28100 C*
28200 C*          IF CMD 3 (*IN97) IS PRESSED, THE ITEM INQUIRY MENU IS          10/14/87
28300 C*          WRITTEN ON THE SCREEN.          10/14/87
28400 C*
28500 C*          IF 'ENTER' IS PRESSED WHILE SCREEN 2 FOR ITEM REQUESTED IS          10/16/87
28600 C*          CURRENTLY DISPLAYED, MORE INFORMATION IS CALCULATED AND          10/16/87
28700 C*          DISPLAYED.          10/16/87
28800 C*
28900 C*          IF 'ENTER' IS PRESSED WHILE SCREEN 3 FOR ITEM REQUESTED IS          10/16/87
29000 C*          CURRENTLY DISPLAYED, THEN THE ITEM INQUIRY MENU IS WRITTEN          03/20/89
29100 C*          TO THE SCREEN.          03/20/89
29200 C*
29300 C*****
29400 C* 8
29500 C          ITMRTN          TAG                10/13/87
29600 C          *IN99          CABEQ '1'          END          JOB ENDS          10/13/87
29700 C          *IN98          IFEQ '1'          B001          10/13/87
29800 C          WRITECIMENU          MAIN MENU          001          10/13/87
29900 C          GOTO READRQ          001          10/13/87
30000 C          END          E001          10/13/87
30100 C          *IN97          IFEQ '1'          CMD 3 ?          B001          10/13/87
30200 C          RECID          IFEQ 'ITMSC2 '          ITM SCR 2 ?          B002          10/13/87

```

Figure 11-21 (Part 5 of 12). Source Program Example — RSFMUL (System-Supplied Formats)

```

30300 C          WRITEITMMNU          YES, THEN ITS          002 10/13/87
30400 C          GOTO READRQ          ITEM MENU          002 10/13/87
30500 C          END                    E002 10/13/87
30600 C          END                    E001 10/13/87
30700 C          RECID  IFEQ 'ITMSC3 '  ITM SCR 3 ?          B001 10/13/87
30800 C          WRITEITMMNU          YES, THEN ITS          001 10/13/87
30900 C          GOTO READRQ          ITEM MENU          001 10/13/87
31000 C          END                    E001 10/13/87
31100 C          SLSTM  SUB  CSTTM  PROFM 92  PROFIT MONTH          10/13/87
31200 C          MULT 100  PROFM          10/13/87
31300 C          SLSTM  COMP 0          46          3          10/13/87
31400 C          N46  PROFM  DIV  SLSTM  PROFM  PROFIT PCT          10/13/87
31500 C          QTYLST MULT PR01  LOSTS  LOST SALES          10/13/87
31600 C          MOVE SLSTM  SLSM          10/13/87
31700 C          MOVE SLSTY  SLSY          10/13/87
31800 C          MOVE CSTTM  CSTM          10/13/87
31900 C          MOVE PROFM  PROFIT          10/13/87
32000 C          MOVE CSTTY  CSTY          10/13/87
5738R61 V2R1M0 910524  IBM AS/400 RPG/400  ICFLIB/RSFMUL 10/03/90 14:55:01 Page 9
SEQUENCE NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE IND DO LAST PAGE PROGRAM
32100 C          WRITEITMSC3          ITEM DTL 2          10/13/87
32200 C          GOTO READRQ          10/13/87
32300 C*****
32400 C*
32500 C*          CUSTOMER INQUIRY          10/13/87
32600 C*          10/13/87
32700 C*          THE REQUEST FROM THE CUSTOMER INQUIRY MENU IS PROCESSED.          10/13/87
32800 C*          10/13/87
32900 C*          IF CMD 1 (*IN99) IS PRESSED, THE PROGRAM IS ENDED.          10/13/87
33000 C*          10/13/87
33100 C*          IF CMD 2 (*IN98) IS PRESSED, THE CUSTOMER INQUIRY IS ENDED,          10/13/87
33200 C*          AND THE MAIN MENU (CIMENU) IS WRITTEN TO THE SCREEN.          10/13/87
33300 C*          10/13/87
33400 C*          IF A CUSTOMER NUMBER IS ENTERED, THE CUSTOMER INQUIRY          10/13/87
33500 C*          REQUEST IS SENT TO THE REMOTE SYSTEM.          10/13/87
33600 C*          10/13/87
33700 C*          A READ TO THE PROGRAM DEVICE IS ISSUED TO RECEIVE THE          03/20/89
33800 C*          INFORMATION FROM THE TARGET PROGRAM.          10/13/87
33900 C*          10/13/87
34000 C*          IF AN ERROR OCCURS, THE ERROR IS PRINTED AND THE JOB IS          10/13/87
34100 C*          ENDED.          10/13/87
34200 C*          10/13/87
34300 C*****
34400 C* 9          03/20/89
34500 C          DTLIN  TAG          10/13/87
34600 C          *IN99  CABEQ'1'  END          JOB ENDS          10/13/87
34700 C          *IN98  IFEQ '1'          B001 10/13/87
34800 C          WRITECIMENU          MAIN MENU          001 10/13/87
34900 C          GOTO READRQ          001 10/13/87
35000 C          END                    E001 10/13/87
35100 C          EVDTL  TAG          10/13/87
35200 C          MOVE CUSTNO  ITEMNO          10/13/87
35300 C          MOVEV 'ICF00 ' CMID          10/13/87
35400 C          EXCPTITEMRQ          SEND CUST #          10/13/87
35500 C          MAJCOD  CABGE'04'  ERROR  ERROR RTN          10/13/87
35600 C          TRY88  TAG          10/13/87
35700 C          MOVEV ' ' CMID          10/13/87
35800 C          MOVEV ' '  RECCUS  RESET RECID          10/13/87
35900 C          SETOF          88          1          10/13/87
36000 C          READ CMNFIL          8810REC CUS INF          2 3          10/13/87
36100 C          88  EXSR ERRCHK          FILE ERR?          10/13/87
36200 C          MAJMIN  CABGE'0300'  EVDTL  NODATATRYAGN          10/13/87
36300 C          N80  EXCPTNOTCUS          RECD NOT FD?          10/13/87
36400 C          RECCUS  CABNE'C'  RECERR  PRINT MSG          10/13/87
36500 C          SETOF          80          1          10/13/87
36600 C*****
36700 C*          10/13/87
36800 C*          PROCESS CUSTOMER INFORMATION          10/13/87
36900 C*          10/13/87
37000 C*          THE CUSTOMER DATA RECEIVED FROM THE TARGET PROGRAM IS          03/20/89
37100 C*          PROCESSED. IF CUSTOMER NUMBER IS ZERO OR LESS, IT IS AN          03/20/89
37200 C*          INVALID REQUEST AND THE MAIN MENU IS WRITTEN TO THE SCREEN.          03/20/89
37300 C*          10/14/87
37400 C*          WHEN THE RCVTRNRND INDICATOR(*IN90) IS RECEIVED, THE          03/20/89
5738R61 V2R1M0 910524  IBM AS/400 RPG/400  ICFLIB/RSFMUL 10/03/90 14:55:01 Page 10
SEQUENCE NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE IND DO LAST PAGE PROGRAM
37500 C*          CUSTOMER INFORMATION IS WRITTEN TO THE SCREEN.          03/20/89
37600 C*          10/14/87

```

Figure 11-21 (Part 6 of 12). Source Program Example — RSFMUL (System-Supplied Formats)


```

37700 C*      IF DURING THE READ OPERATION AN ERROR IS RECEIVED,          10/13/87
37800 C*      CONTROL GOES TO THE ERROR ROUTINE TO END THE JOB.          10/13/87
37900 C*                                                                 10/13/87
38000 C*****                                                                 10/13/87
38100 * 10                                                                 10/13/87
38200 C          DTOUT      TAG                                          10/13/87
38300 C          CUSTNO    IFEQ 000000                                  B001 10/13/87
38400 C          SETOF                                66          3 001 10/13/87
38500 C          WRITECIMENU                                MAIN MENU      001 10/13/87
38600 C          GOTO READRQ                                001 10/13/87
38700 C          END                                          E001 10/13/87
38800 C          MOVE CUSTNO  CUSTN                                10/13/87
38900 C          MOVELDNAME CNAME                                08/08/89
39000 C          MOVE DLSTOR DLSTR                                10/13/87
39100 C          MOVE DSLSTM DSLSM                                10/13/87
39200 C          MOVE DSPM01 DSPM1                                10/13/87
39300 C          MOVE DSPM02 DSPM2                                10/13/87
39400 C          MOVE DSTTYD DSTYD                                10/13/87
39500 C          MOVE IDEPT  DEPT                                  10/13/87
39600 C          WRITEDLSCR                                DETAIL INFO 10/13/87
39700 C          GOTO READRQ                                10/13/87
39800 C*****                                                                 10/13/87
39900 C*                                                                 10/13/87
40000 C*      THIS ROUTINE HANDLES THE USER'S REQUEST FOLLOWING THE      10/14/87
40100 C*      DISPLAY OF THE CUSTOMER INFORMATION.  CMD KEY 1 WILL END    03/20/89
40200 C*      THE JOB, CMD KEY 2 WILL DISPLAY THE MAIN MENU, AND "ENTER" 03/20/89
40300 C*      WILL BRING UP THE CUSTOMER INQUIRY MENU.                  03/20/89
40400 C*                                                                 10/13/87
40500 C*****                                                                 10/13/87
40600 * 11                                                                 10/13/87
40700 C          DTLRTN    TAG                                          10/13/87
40800 C          *IN99    CABEQ'1'      END          JOB ENDS          10/13/87
40900 C          *IN98    IFEQ '1'                                  B001 10/13/87
41000 C          WRITECIMENU                                MAIN MENU      001 10/13/87
41100 C          GOTO READRQ                                001 10/13/87
41200 C          END                                          E001 10/13/87
41300 C          WRITEDLMNU                                CUSTOMER INQ 10/13/87
41400 C          GOTO READRQ                                10/13/87
41500 C*****                                                                 10/13/87
41600 C*                                                                 10/13/87
41700 C*      WHEN AN I/O OPERATION ERROR IS DETECTED, A MESSAGE IS PRINTED 03/20/89
41800 C*      AND THE TRANSACTION AND SESSION ARE ENDED FOR EACH OF THE 03/20/89
41900 C*      REMOTE SYSTEMS.                                           03/20/89
42000 C*                                                                 10/13/87
42100 C*****                                                                 10/13/87
42200 * 12                                                                 10/13/87
42300 C          RECERR    TAG                                          10/13/87
42400 C          EXCPTRECER                                WRONG RECID 10/13/87
42500 C          GOTO END          END PROGRAM                    10/13/87
42600 C          ERROR    TAG                                          10/13/87
42700 C          EXCPTMERR                                10/13/87
42800 C          END      TAG                                          10/13/87
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RFSMUL 10/03/90 14:55:01 Page 11
SEQUENCE          NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...*  IND  DO  LAST  PAGE  PROGRAM
NUMBER          *...1...+...2...+...3...+...4...+...5...+...6...+...7...*  USE  NUM  UPDATE LINE  ID
42900 C          MOVE 'ICF00 'CMID                                10/13/87
43000 C          EXCPTDETACH                                DET 1ST TRANS 10/13/87
43100 C          MOVE 'ICF01 'CMID                                10/13/87
43200 C          EXCPTDETACH                                DET 2ND TRANS 10/13/87
43300 C          MOVE 'ICF02 'CMID                                10/13/87
43400 C          EXCPTDETACH                                DET 3RD TRANS 10/13/87
43500 C          MOVE 'ICF03 'CMID                                10/13/87
43600 C          EXCPTDETACH                                DET 4TH TRANS 10/13/87
43700 C          ABORT    TAG                                          10/13/87
43800 C          'ICF00 'REL CMNFIL          86 REL 1ST SESS      2 10/13/87
43900 C          'ICF01 'REL CMNFIL          86 REL 2ND SESS      2 10/13/87
44000 C          'ICF02 'REL CMNFIL          86 REL 3RD SESS      2 10/13/87
44100 C          'ICF03 'REL CMNFIL          86 REL 4TH SESS      2 10/13/87
44200 C          FORCE    TAG                                          10/13/87
44300 C          SETON                                LR          1 10/13/87
44400 C          RETRN                                10/13/87
44500 C*****                                                                 10/13/87
44600 C*                                                                 10/13/87
44700 C*      THIS SUBROUTINE IS CALLED TO EVOKE THE TARGET PROGRAM. THE 03/20/89
44800 C*      SAME TARGET PROGRAM (ICFLIB/RTDMULCL) IS EVOKED AT FOUR    03/20/89
44900 C*      DIFFERENT REMOTE SYSTEMS.  THE PROGRAM DEVICE IDENTIFIES 03/20/89
45000 C*      WHICH SESSION SHOULD BE EVOKED.  THE PROGRAM DEVICE WAS    03/20/89
45100 C*      SPECIFIED IN CMID PRIOR TO CALLING THIS ROUTINE.          03/20/89
45200 C*                                                                 10/13/87
45300 C*****                                                                 10/13/87

```

Figure 11-21 (Part 7 of 12). Source Program Example — RFSMUL (System-Supplied Formats)

```

45400 * 13 10/13/87
45500 C EVKSR BEGSR 10/13/87
45600 C EXCPTVOKE EVOKE TARGET 10/13/87
45700 C ENDSR 10/13/87
45800 C***** 10/13/87
45900 C* 10/13/87
46000 C* THIS SUBROUTINE IS CALLED TO PERFORM FURTHER CHECKS ON FILE 03/20/89
46100 C* ERRORS RESULTING FROM THE READ OPERATION ISSUED TO THE PRO- 03/20/89
46200 C* GRAM DEVICE. THIS ROUTINE CHECKS FOR THE TIME OUT INDICATION. 03/20/89
46300 C* IF THERE IS A TIME OUT, THEN A MESSAGE IS SENT TO THE USER'S 03/20/89
46400 C* DISPLAY SCREEN REQUESTING ACTION, OTHERWISE PROGRAM ENDS. 10/16/87
46500 C* 10/13/87
46600 C***** 10/13/87
46700 * 14 10/13/87
46800 C ERRCHK BEGSR 10/13/87
46900 C MAJMIN IFEQ '0310' TIMER EXPD? B001 10/13/87
47000 C CHKAGN TAG 001 10/13/87
47100 C WRITETIMOUT DISPLAY MSG 001 10/13/87
47200 C READ DSPFIL 87READ REPLY 3 001 10/13/87
47300 C 88 TIMRSP CABEQ'1' TRY88 CUST INQUIR 001 10/13/87
47400 C 89 TIMRSP CABEQ'1' TRY89 ITEM INQUIR 001 10/13/87
47500 C TIMRSP IFEQ '2' END PROGRAM B002 10/13/87
47600 C EXCPTEOS END SESSION 002 10/13/87
47700 C GOTO FORCE END PROGRAM 002 10/13/87
47800 C END E002 10/13/87
47900 C GOTO CHKAGN ASK AGAIN 001 10/13/87
48000 C END E001 10/13/87
48100 C MAJMIN CABEQ'0300' ERRESR TRN/NODATA 10/13/87
48200 C MAJMIN CABEQ'0000' ERRESR TRN W/DATA 10/13/87
5738R61 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSFMUL 10/03/90 14:55:01 Page 12
SEQUENCE IND DO LAST PAGE PROGRAM
NUMBER *...1....+...2....+...3....+...4....+...5....+...6....+...7...* USE NUM UPDATE LINE ID
48300 C GOTO ERROR ABEND 10/13/87
48400 C ERRESR TAG 10/13/87
48500 C ENDSR 10/13/87
48600 C***** 10/14/87
48700 C* 10/14/87
48800 C* THIS IS THE PROGRAM ERROR SUBROUTINE THAT RECEIVES CONTROL 03/20/89
48900 C* WHEN AN ERROR OCCURS AFTER AN I/O OPERATION IS ISSUED TO THE 03/20/89
49000 C* PROGRAM DEVICE AND THERE IS A NON-ZERO VALUE IN THE RPG 03/20/89
49100 C* STATUS FIELD (ERR). THIS ROUTINE CHECKS FOR STATUS VALUES 03/20/89
49200 C* THAT RELATE TO ICF OPERATIONS. IF THE PROGRAM DEVICE 10/03/90
49300 C* IS ALREADY ACQUIRED, THE ERROR IS IGNORED, OTHERWISE THE 03/20/89
49400 C* PROGRAM IS TERMINATED. 03/20/89
49500 C* 10/14/87
49600 C***** 10/14/87
49700 * 15 10/14/87
49800 C *PSSR BEGSR 10/14/87
49900 C MOVE ' ' RETURN 6 DEFAULT 10/14/87
50000 C ERR CABEQ01285 ENDPSR ALREADY ACQ? 10/14/87
50100 C MOVE '*CANCL' RETURN JOB ENDS 10/14/87
50200 C ENDPSR ENDSRRETURN BACK TO MAIN 10/14/87
50300 C***** 10/13/87
50400 OQPRINT E 1 MMERR 10/13/87
50500 0 21 'COMMUNICATION ERROR.' 10/13/87
50600 0 34 'MAJOR/MINOR:' 10/13/87
50700 0 MAJCOD 37 10/13/87
50800 0 38 '/' 10/13/87
50900 0 MINCOD 40 10/13/87
51000 0 49 'FORMAT:' 10/13/87
51100 0 FMTNM 60 10/13/87
51200 0 69 'PGMDEV:' 10/13/87
51300 0 CMID 80 10/13/87
51400 0 E 1 RECER 10/13/87
51500 0 22 'UNMATCH RECORD FORMAT' 10/13/87
51600 0 34 '-JOB ENDED.' 10/13/87
51700 0 E 1 NOTITM 10/13/87
51800 0 21 'NOT ITEM RECD-' 10/13/87
51900 0 ITEMNO 28 10/13/87
52000 0 29 '/' 10/13/87
52100 0 DESC 60 10/13/87
52200 0 E 1 NOTCUS 10/13/87
52300 0 21 'NOT CUST RECD-' 10/13/87
52400 0 CUSTNO 28 10/13/87
52500 0 29 '/' 10/13/87
52600 0 DNAME 60 10/13/87
52700 OCMNFIL E EVOKE 10/13/87
52800 0 K8 '$$EVOKNI' 10/13/87
52900 0 8 'RTFMULCL' 10/13/87
53000 0 32 'ICFLIB ' 10/13/87

```

Figure 11-21 (Part 8 of 12). Source Program Example — RSFMUL (System-Supplied Formats)

```

53100 0      E      ITEMRQ      10/13/87
53200 0      K6 '$$SEND'      10/13/87
53300 0      4 '0000'      10/13/87
53400 0      ITEMNO 10      10/13/87
5738RG1 V2R1M0 910524      IBM AS/400 RPG/400      ICFLIB/RSFMUL      10/03/90 14:55:01      Page      13
SEQUENCE      IND      DO      LAST      PAGE      PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE      NUM      UPDATE      LINE      ID
53500 0      E      DETACH      10/13/87
53600 0      K8 '$$SENDET'      10/13/87
53700 0      4 '0000'      10/13/87
53800 0      E      EOS      10/13/87
53900 0      K5 '$$EOS'      10/13/87
54000 0      4 '0000'      10/13/87
* 6103      54001      OVERFLOW INDICATOR OA ASSIGNED TO FILE QPRINT.
H000000      OUTPUT FIELDS FOR RECORD CIMENU FILE DSPFIL FORMAT CIMENU.
H000000      MENU FOR INQUIRY
I000000      OUTPUT FIELDS FOR RECORD DTLMNU FILE DSPFIL FORMAT DTLMNU.
I000000      CUSTOMER INQUIRY SCREEN 1
J000000      OUTPUT FIELDS FOR RECORD DTLSCR FILE DSPFIL FORMAT DTLSCR.
J000000      CUSTOMER INQUIRY SCR. #2
J000001      CUSTN      6      CHAR      6
J000002      DEPT      9      ZONE      3,0
J000003      DLSTR      15      ZONE      6,0
J000004      DSLSM      24      ZONE      9,0
J000005      DSPM1      33      ZONE      9,0
J000006      DSPM2      42      ZONE      9,0
J000007      DSPM3      51      ZONE      9,0
J000008      DSTYD      62      ZONE      11,0
J000009      CNAME      67      CHAR      5
K000000      OUTPUT FIELDS FOR RECORD ITMMNU FILE DSPFIL FORMAT ITMMNU.
K000000      ITEM INQUIRY SCREEN ONE
L000000      OUTPUT FIELDS FOR RECORD ITMSC2 FILE DSPFIL FORMAT ITMSC2.
L000000      ITEM INQUIRY SCREEN TWO
L000001      DSC      30      CHAR      30
L000002      QAVAIL      37      ZONE      7,0
L000003      QTYH      44      ZONE      7,0
L000004      QTYO      51      ZONE      7,0
L000005      QTYB      58      ZONE      7,0
L000006      UNT      60      CHAR      2
L000007      PR1      67      ZONE      7,2
L000008      PR5      74      ZONE      7,0
L000009      UFR      79      ZONE      5,2
M000000      OUTPUT FIELDS FOR RECORD ITMSC3 FILE DSPFIL FORMAT ITMSC3.
M000000      ITEM INQUIRY SCREEN 3
M000001      SLSM      9      ZONE      9,2
M000002      SLSY      26      ZONE      11,2
M000003      CSTM      29      ZONE      9,2
M000004      CSTY      40      ZONE      11,2
M000005      PROFIT      45      ZONE      5,2
M000006      LOSTS      54      ZONE      9,2
N000000      OUTPUT FIELDS FOR RECORD TIMOUT FILE DSPFIL FORMAT TIMOUT.
N000000      TIME OUT SCREEN
***** END OF SOURCE *****
Additional Diagnostic Messages
* 7089      4500      RPG PROVIDES SEPARATE INDICATOR AREA FOR FILE CMNFIL.
5738RG1 V2R1M0 910524      IBM AS/400 RPG/400      ICFLIB/RSFMUL      10/03/90 14:55:01      Page      14
Cross Reference
File and Record References:
FILE/RCD      DEV/RCD      REFERENCES (D=DEFINED)
01      CMNFIL      WORKSTN      4500D      6500      7700      12900      13000
      13100      13200      23000      36000      43800
      43900      44000      44100      52700      53100
      53500      53800
      $$EOS      53900
      $$EVOKNI      52800
      $$SEND      53200
      $$SENDET      53600
02      DSPFIL      WORKSTN      5000D      15600      47200
      CIMENU      5000D      A000000      14200      16300      20900
      29800      34800      38500      41000      H000000
      DTLMNU      5000D      B000000      18200      41300      I000000
      DTLSCR      5000D      C000000      39600      J000000
      ITMMNU      5000D      D000000      18000      25100      30300
      30800      K000000
      ITMSC2      5000D      E000000      26600      L000000
      ITMSC3      5000D      F000000      32100      M000000
      TIMOUT      5000D      G000000      47100      N000000
03      QPRINT      PRINTER      5200D      50400      51400      51700      52200
      54001

```

Figure 11-21 (Part 9 of 12). Source Program Example — RSFMUL (System-Supplied Formats)

Field References:

FIELD	ATTR	REFERENCES (M=MODIFIED D=DEFINED)				
*IN97	A(1)	A000001 B000001 C000001 D000001 E000001				
		F000001 G000001 30100				
*IN98	A(1)	A000002 B000002 C000002 D000002 E000002				
		F000002 G000002 20800 29700 34700				
		40900				
*IN99	A(1)	A000003 B000003 C000003 D000003 E000003				
		F000003 G000003 17800 20700 29600				
		34600 40800				
*PSSR	BEGSR	4500 49800D				
* 7031 ABORT	TAG	43700D				
* 7031 BEGIN	TAG	12800D				
CHKAGN	TAG	4700D 47900				
CMID	A(10)	10600D 13300M 13500M 13700M 13900M				
		21600M 21900M 22200M 22900M 35300M				
		35700M 42900M 43100M 43300M 43500M				
		51300				
CNAME	A(5)	38900M J000009D				
CSTM	P(9,2)	31800M M000003D				
CSTTM	P(9,2)	9100D 31100 31800				
CSTTY	P(11,2)	9200D 32000				
5738RG1 V2R1M0	910524	IBM AS/400 RPG/400	ICFLIB/RFSMUL	10/03/90	14:55:01	Page 15
CSTY	P(11,2)	32000M M000004D				
CUSTN	A(6)	38800M J000001D				
CUSTNO	P(6,0)	6700D B000004D 35200 38300 38800				
		52400				
DEPT	P(3,0)	39500M J000002D				
DESC	A(30)	8000D 25800 52100				
DETACH	EXCPT	43000 43200 43400 43600 53500				
DLSTOR	A(6)	6900D 39000				
DLSTR	P(6,0)	39000M J000003D				
DNAME	A(30)	6800D 38900 52600				
DSC	A(30)	25800M L000001D				
DSLSTM	P(9,0)	39100M J000004D				
DSLSTM	A(9)	7000D 39100				
DSPM01	A(9)	7100D 39200				
DSPM02	A(9)	7200D 39300				
* 7031 DSPM03	A(9)	7300D				
DSPM1	P(9,0)	39200M J000005D				
DSPM2	P(9,0)	39300M J000006D				
DSPM3	P(9,0)	J000007D				
DSTTYD	A(11)	7400D 39400				
DSTYD	P(11,0)	39400M J000008D				
DTLIN	TAG	16100 34500D				
DTLRTN	TAG	16200 40700D				
* 7031 DTOUT	TAG	38200D				
END	TAG	17800 20700 29600 34600 40800				
		42500 42800D				
ENDPSR	ENDSR	50900 50200D				
EOS	EXCPT	47600 53800				
ERR	Z(5,0)	10300D 50000				
ERRCHK	BEGSR	23100 36100 46800D				
ERRRESR	TAG	48100 48200 48400D				
ERROR	TAG	22500 23500 35500 42600D 48300				
EVDTL	TAG	35100D 36200				
EVKSR	BEGSR	13400 13600 13800 14000 45500D				
EVOKE	EXCPT	45600 52700				
* 7031 FILL01	A(240)	9800D				
* 7031 FILL02	A(145)	10000D				
* 7031 FILL03	A(240)	10400D				
* 7031 FILL04	A(145)	11100D				
* 7031 FILL10	A(11)	9500D				
* 7031 FILL20	A(57)	7600D				
FMTNM	A(8)	10500D 51100				
FORCE	TAG	44200D 47700				
IDEPT	A(3)	7500D 39500				
IODS	DS(415)	5000 9700D				
IOFB	DS(415)	4500 10100D				
ITEMNO	P(6,0)	7900D D000004D 21200 21300 21400				
		25000 35200M 51900 53400				
ITEMRQ	EXCPT	22400 35400 53100				
ITMIN	TAG	15800 20600D 23200				
* 7031 ITMOUT	TAG	24900D				
ITMRTN	TAG	15900 16000 29500D				
* 7031 LOC	A(8)	10200D				
* 7031 LOS	P(9,2)	9400D				
LOSTS	P(9,2)	31500M M000006D				
* 7031 MAIN	TAG	14100D				
MAJCOD	A(2)	10800D 22500 23500 35500 50700				

Figure 11-21 (Part 10 of 12). Source Program Example — RSFMUL (System-Supplied Formats)

Job ID	Program	Version	Job ID	System	Language	ICFLIB/RSFMUL	Date	Time	Page
5738R61	V2R1M0	910524		IBM AS/400	RPG/400		10/03/90	14:55:01	Page 16
	MAJMIN	A(4)	107000	23200	36200	46900 48100			
			48200						
	MENU	TAG	15700	177000					
	MINCOD	A(2)	109000	50900					
	MMERR	EXCPT	42700	50400					
	NOTCUS	EXCPT	36300	52200					
	NOTITM	EXCPT	23300	51700					
	OPTION	A(1)	A000004D	17900					
* 7031	PR0	P(5,2)	93000						
	PROFIT	P(5,2)	31900M	M000005D					
	PROFM	P(9,2)	311000	31200M	31400	31400M 31900			
	PR01	P(7,2)	86000	26300	31500				
	PR05	P(6,0)	87000	26400					
	PR1	P(7,2)	26300M	L000007D					
	PR5	P(7,0)	26400M	L000008D					
	QAVAIL	P(7,0)	254000	25500M	25600M	25700M L000002D			
	QTYB	P(7,0)	26100M	L000005D					
	QTYB0	P(7,0)	84000	25700	26100				
	QTYH	P(7,0)	26000M	L000003D					
	QTYLST	P(7,0)	81000	31500					
	QTYO	P(7,0)	25900M	L000004D					
	QTYOH	P(7,0)	82000	25500	26000				
	QTY00	P(7,0)	83000	25600	25900				
	READRQ	TAG	155000	16400	18400	21000 25200			
			26700	29900	30400	30900 32200			
			34900	38600	39700	41100 41400			
	RECCUS	A(1)	66000	35800M	36400				
	RECER	EXCPT	42400	51400					
	RECERR	TAG	23600	36400	42300D				
	RECID	A(8)	99000	15700	15800	15900 16000			
			16100	16200	30200	30700			
* 7031	RECID2	A(8)	110000						
	RECITM	A(1)	78000	22800M	23600				
	RETURN	A(6)	499000	50100M	50200				
	SLSM	P(9,2)	31600M	M000001D					
	SLSTM	P(9,2)	89000	31100	31300	31400 31600			
	SLSTY	P(11,2)	90000	31700					
	SLSY	P(11,2)	31700M	M000002D					
	TIMRSP	A(1)	6000004D	47300	47400	47500			
	TRY88	TAG	356000	47300					
	TRY89	TAG	226000	47400					
	UFR	P(5,2)	26500M	L000009D					
	UFRT	P(5,2)	88000	26500					
	UNIT	A(2)	85000	26200					
	UNT	A(2)	26200M	L000006D					
	XICF01	TAG	21200	215000					
	XICF02	TAG	21300	218000					
	XICF03	TAG	21400	221000					
	XITMIN	TAG	21700	22000	22300D				
	'	LITERAL	22900	35700					
	'	LITERAL	49900						
	'	LITERAL	22800	35800					
	'*CANCL'	LITERAL	50100						
	'C'	LITERAL	36400						
	'CIMENU'	LITERAL	15700						
	'DTLMNU'	LITERAL	16100						
	'DTLSCR'	LITERAL	16200						
5738R61	V2R1M0	910524		IBM AS/400	RPG/400		10/03/90	14:55:01	Page 17
	'1'	LITERAL	23600						
	'ICF00'	LITERAL	12900	13300	35300	42900 43800			
	'ICF01'	LITERAL	13000	13500	21600	43100 43900			
	'ICF02'	LITERAL	13100	13700	21900	43300 44000			
	'ICF03'	LITERAL	13200	13900	22200	43500 44100			
	'ITMMNU'	LITERAL	15800						
	'ITMSC2'	LITERAL	15900	30200					
	'ITMSC3'	LITERAL	16000	30700					
	'0000'	LITERAL	48200						
	'0300'	LITERAL	23200	36200	48100				
	'0310'	LITERAL	46900						
	'04'	LITERAL	22500	23500	35500				
	'1'	LITERAL	17800	17900	20700	20800 29600			
			29700	30100	34600	34700 40800			
			40900	47300	47400				
	'2'	LITERAL	47500						
	0	LITERAL	25400	31300					
	000000	LITERAL	25000	38300					
	01285	LITERAL	50000						
	100	LITERAL	31200						
	399999	LITERAL	21200						
	699999	LITERAL	21300						

Figure 11-21 (Part 11 of 12). Source Program Example — RSFMUL (System-Supplied Formats)

```

899999 LITERAL 21400
Indicator References:
INDICATOR REFERENCES (M=MODIFIED D=DEFINED)
*IN A000001 A000002 A000003 B000001 B000002 B000003
C000001 C000002 C000003 D000001 D000002 D000003
E000001 E000002 E000003 F000001 F000002 F000003
G000001 G000002 G000003 17800 20700 20800
29600 29700 30100 34600 34700 40800
40900
LR 44300M
OA 5200D 54001
* 7031 10 23000M 36000M
46 31300M 31400
* 7031 66 38400M
80 6500M 36300 36500M
81 7700M 23300 23400M
* 7031 86 43800M 43900M 44000M 44100M
* 7031 87 47200M
88 15600M 35900M 36000M 36100 47300
89 22700M 23000M 23100 47400
97 A000001 B000001 C000001 D000001 E000001 F000001
G000001 30100
98 A000002 B000002 C000002 D000002 E000002 F000002
G000002 20800 29700 34700 40900
99 A000003 B000003 C000003 D000003 E000003 F000003
G000003 17800 20700 29600 34600 40800
***** E N D O F C R O S S R E F E R E N C E *****
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSFMUL 10/03/90 14:55:01 Page 18
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSFMUL 10/03/90 14:55:01 Page 19
Message Summary
* QR6103 Severity: 00 Number: 1
Message . . . . : No Overflow Indicator is specified but an
indicator is assigned to a file and automatic skip to 6 is
generated.
* QR67031 Severity: 00 Number: 20
Message . . . . : The Name or indicator is not referenced.
* QR67089 Severity: 00 Number: 1
Message . . . . : The RPG provides Separate-Indicator area for
file.
***** E N D O F M E S S A G E S U M M A R Y *****
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RSFMUL 10/03/90 14:55:01 Page 20
Final Summary
Message Count: (by Severity Number)
TOTAL 00 10 20 30 40 50
22 22 0 0 0 0 0
Program Source Totals:
Records . . . . . : 540
Specifications . . . . . : 296
Table Records . . . . . : 0
Comments . . . . . : 244
PRM has been called.
Program RSFMUL is placed in library ICFLIB. 00 highest Error-Severity-Code.
***** E N D O F C O M P I L A T I O N *****

```

Figure 11-21 (Part 12 of 12). Source Program Example — RSFMUL (System-Supplied Formats)

Target Program Multiple-Session Inquiry

(Example II): The following describes a target program for the multiple-session inquiry.

Program Files: The RPG/400 multiple-session example target program uses the following files:

CFILE

An ICF file used to send records to and receive records from the source program.

PFILE

A database file used to retrieve the requested information to send to the source program.

QPRINT

A printer file used to print error messages resulting from communications errors.

DDS Source: The DDS source for the ICF file (CFILE) is illustrated in Figure 11-22 on page 11-63.

```

5714PW1 R01M00 910524          SEU SOURCE LISTING          10/14/87 17:20:35          PAGE 1
SOURCE FILE . . . . . QICFPUB/ICFLIB
MEMBER . . . . . CFILE
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100  A*****
200  A*
300  A*          ICF FILE          *
400  A*          USED IN TARGET MULTIPLE SESSION PROGRAM          *
500  A*
600  A*****
700  A          INDARA
800  A 05          RQSWRT
900  A 10          ALWVRT
1000 A          INDTXT(10 '10 END TRANS.')
1100 A 15          EOS
1200 A 20          FAIL
1300 A          INDTXT(20 '20 F ABORT ST')
1400 A          RCVFAIL(25 'RECEIVED FAIL')
1500 A 30          DETACH
1600 A          INDTXT(30 '30>DETACH TGT')
1700 A          RCVDETACH(44 'REC'V DETACH')
1800 A          R SNDPART
1900 A          INVITE
2000 A          RECTYP          1
2100 A          ITEMNO          6
2200 A          EDATA          130
2300 A          FILL1          13
2400 A          R RCVPART
2500 A          RECID2          6
2600 A          PARTDS          80
2700 A          FILL4          64
2800 A          R RCVTRND
2900 A          RCVTRNRND(40 'END OF TRN')
          *****
          * * * * * E N D   O F   S O U R C E   * * * * *

```

Figure 11-22. DDS Source for ICF File Used by Target Program Multiple-Session Inquiry

The DDS for the database file (PFILE) is illustrated in Figure 11-23.

```

5714PW1 R01M00 910524          SEU SOURCE LISTING          10/16/87 07:43:14          PAGE 1
SOURCE FILE . . . . . QICFPUB/ICFLIB
MEMBER . . . . . PFILE
SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+... 0
100  A          LIFO
200  A          R DBREC
300  A          RECCUS          1
400  A          DRSEQ          6
500  A          DBDATA          130
600  A          DBFILL          13
700  A          K DBSEQ
          *****
          * * * * * E N D   O F   S O U R C E   * * * * *

```

Figure 11-23. DDS Source for Database File Used by Target Program Multiple Session Inquiry

ICF File Creation and Program Device Entry

Definition: The command needed to create the ICF file is:

```

CRTICFF FILE(ICFLIB/CFILE) SRCFILE(ICFLIB/QICFPUB)
SRCMBR(CFILE) ACQPGMDEV(RQSDEV) TEXT("TARGET ICF FILE
FOR MULTIPLE SESSION PROGRAM")

```

The command needed to define the program device entry is:

```

OVRICFDEVE PGMDEV(RQSDEV) RMTLOCNAME(*REQUESTER)

```

Program Explanation: The following explains the structure of the program examples illustrated in Figure 11-24 on page 11-65 and Figure 11-25 on page 11-69. The ICF file used in the first example is defined by the user, and uses

externally described data formats (DDS). The second example uses the same file, but uses program-described data and system-supplied formats. The reference numbers in the explanation below correspond to the numbers in the program examples.

Although the basic structure of the two examples provided is the same, there are differences because of the way the user-defined formats and the system-supplied formats are used. All output operations to the ICF file in the first example are done using the WRITE operation. All output operations in the ICF file in the second example using system-supplied formats are done using the EXCPT operation.

Differences between the first and second example are described as notes in each of the following descriptions where necessary.

- 1** The file specification defines the files used in the program.

CFILE is the ICF file used to send records to and receive records from the source program.

The files used in the program are implicitly opened at the beginning of the RPG/400 cycle when the program starts.

Note: In the program using system-supplied formats, the input records for CFILE are explicitly coded in the program since CFILE is now described as a program-described file. The system-supplied file QICDMF can be used instead of CFILE. To use QICDMF, specify QICDMF in the file specification, or use an OVRICFF command to change the file name from CFILE to QICDMF.

The continuation lines on the file specification for CFILE define the data structure name — FEEDBK for the feedback area (INFDS). FEEDBK contains the following information, which is used to monitor for error conditions after an I/O operation is issued to CFILE:

- Record format-name (FMTNM)
- Program device name (PGMDEV)
- Major/minor return code (MAJMIN)

- 2** A read operation is issued to the program device to receive an inquiry request from the source program. If an error occurs on the read operation (a major code greater than 03), control passes to **5**.

If a detach indication is received, control goes to **6**. Otherwise, the program goes to **3**. When a detach is received, indicator 44 is set on, as defined by the RCVDETACH keyword in the DDS for the ICF file.

Note: In the program using system-supplied formats, a minor return code of 08 is checked to determine if a detach indication was received. Also, the read operation is issued using file name CFILE in factor 2, whereas in the user-supplied

format example, it is issued using a record format name.

- 3** If a turnaround indication was not received in **2**, the program continues to read the ICF file until the indication is received.

If an error occurs (a major return code greater than 03 is returned from the read operation), the program goes to **5**. Otherwise, the program goes to **4**.

The program also tests to see whether the receive detach indicator (indicator 44) is set. If it is, the program goes to **6**.

Note: In the program using system-supplied formats, a minor return code 00 is checked to determine if change direction occurred and a minor return code of 08 for a detach indication received.

- 4** The program uses the requested number received from the source program to access the record from the database. The information retrieved from the database file (PFILE) is moved into the work area for the ICF file. A write operation is issued to the ICF program device using record format SNDPART. The write operation sends the requested information back to the source program.

If the requested number is not found, zero is propagated into the field.

If an error occurs on the write operation (a major return code greater than 03), control passes to **5**.

If no error occurs on the write, the program goes back to **2**.

Note: In the program using the system-supplied format, the write operation uses the \$\$\$SEND format to send the data.

- 5** When an error in an I/O operation is detected, an EXCPT operation is issued to print an error message saying that an error has occurred on the ICF file. The major/minor return code is also printed.

The program then goes to **6**.

- 6** Control passes here whenever the program has detected a communication error, or received a detach indication from the source program. The last record indicator is set on, which ends the program. CFILE is implicitly closed.

- 7** This subroutine is called for I/O operation errors that are not handled by subroutine **6**. This subroutine checks whether the program device is already acquired when

an acquire operation is requested, and, if so, the second acquire is ignored. Otherwise, the program ends.

```

5738R61 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTDMUL          10/03/90 14:53:44          Page          1
Compiler . . . . . : IBM AS/400 RPG/400
Command Options:
Program . . . . . : ICFLIB/RTDMUL
Source file . . . . . : ICFLIB/QICFPUB
Source member . . . . . : *PGM
Text not available for message RXT0073 file QRPMSG.
Generation options . . . . . : *NOLIST *NOXREF *NOATR *NODUMP *NOOPTIMIZE
Source listing indentation . . . . . : *NONE
SAA flagging . . . . . : *NOFLAG
Generation severity level . . . . . : 9
Print file . . . . . : *LIBL/QSYSPRT
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Text . . . . . : *SRCMBRTXT
Phase trace . . . . . : *NO
Intermediate text dump . . . . . : *NONE
Snap dump . . . . . : *NONE
Codelist . . . . . : *NONE
Ignore decimal data error . . . . . : *NO
Actual Program Source:
Member . . . . . : RTDMUL
File . . . . . : QICFPUB
Library . . . . . : ICFLIB
Last Change . . . . . : 10/03/90 14:50:39
Description . . . . . : RPG Multi-Session example w/DDS (target)
5738R61 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTDMUL          10/03/90 14:53:44          Page          2
SEQUENCE          IND          DO          LAST          PAGE          PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE          NUM          UPDATE          LINE          ID
          S o u r c e          L i s t i n g
100 H*****
200 H*
300 H* THIS PROGRAM WILL HANDLE THE REQUEST FOR EITHER A CUSTOMER
400 H* NUMBER OR AN ITEM NUMBER. THIS IS ACCOMPLISHED BY MAKING
500 H* THE DATABASE FILE STRUCTURE (KEY LENGTH, KEY POSITION, RECORD
600 H* LENGTH, RECORD SIZE, ETC.) THE SAME FOR BOTH FILES WITH ONLY
700 H* THE RECORD CONTENTS DIFFERENT.
800 H*
900 H* THIS PROGRAM ENDS WHEN A DETACH REQUEST IS RECEIVED FROM
1000 H* THE SOURCE PROGRAM.
1100 H*
1200 H*****
          H          *****
1300 FCFILE CF E          WORKSTN          10/13/87
1400 F          KINFDS FEEDBK          10/13/87
1500 F          KINFSR *PSSR          10/14/87
          RECORD FORMAT(S): LIBRARY ICFLIB FILE CFILE.
          EXTERNAL FORMAT SNDPART RPG NAME SNDPART
          EXTERNAL FORMAT RCVPART RPG NAME RCVPART
          EXTERNAL FORMAT RCVTRND RPG NAME RCVTRND
1600 FPFILF IF E          K          DISK          10/13/87
          RECORD FORMAT(S): LIBRARY ICFLIB FILE PFILE.
          EXTERNAL FORMAT DBREC RPG NAME DBREC
1700 FQPRINT 0 F 132          PRINTER          10/13/87
1800 * 1          03/26/89
A000000 INPUT FIELDS FOR RECORD SNDPART FILE CFILF FORMAT SNDPART.
A000001          1 1 RECTYP
A000002          2 7 ITEMNO
A000003          8 137 EDATA
A000004          138 150 FILL1
B000000 INPUT FIELDS FOR RECORD RCVPART FILE CFILF FORMAT RCVPART.
B000001          1 6 RECID2
B000002          7 86 PARTDS
B000003          87 150 FILL4
C000000 INPUT FIELDS FOR RECORD RCVTRND FILE CFILF FORMAT RCVTRND.
D000000 INPUT FIELDS FOR RECORD DBREC FILE PFILE FORMAT DBREC.
D000001          1 1 RECCUS
D000002          2 7 DBSEQ

```

Figure 11-24 (Part 1 of 4). Target Program Example — RTDMUL (User-Defined Formats)

```

D000003          8 137 DBDATA
D000004          138 150 DBFILL
1900 IFEEDBK     DS                                10/13/87
2000 I                                *ROUTINE LOC 10/14/87
2100 I                                *STATUS ERR 10/14/87
2200 I                                38 45 FMTNM 10/13/87
2300 I                                273 282 PGMDEV 10/13/87
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTDMUL 10/03/90 14:53:44 Page 3
SEQUENCE
NUMBER *...1....+...2....+...3....+...4....+...5....+...6....+...7...* USE NUM LAST PAGE PROGRAM
                UPDATE LINE ID
2400 I                                401 404 MAJMIN 10/13/87
2500 I                                401 402 MAJCOD 10/13/87
2600 I                                403 404 MINCOD 10/13/87
2700 C*****
2800 C*
2900 C* READ THE REQUEST FROM THE SOURCE PROGRAM. INDICATOR 40 10/13/87
3000 C* INDICATES RCVTRNRND OCCURRED. INDICATOR 44 INDICATES THAT 10/13/87
3100 C* DETACH HAS BEEN RECEIVED. 10/13/87
3200 C* 10/13/87
3300 C* INDICATOR 99 WILL BE TURNED ON FOR "I/O ERRORS" THEREBY 10/13/87
3400 C* PREVENTING THE RPG DEFAULT ERROR HANDLER FROM BEING CALLED. 10/13/87
3500 C* THIS IS NECESSARY TO ALLOW THE PROGRAM TO PROCESS THE ICF 10/03/90
3600 C* MAJOR/MINOR RETURN CODE. THIS PROGRAM CHECKS FOR ERRORS ON 10/13/87
3700 C* EVERY ICF FILE OPERATION. A MAJOR CODE GREATER THAN 03 10/03/90
3800 C* INDICATES AN ERROR. 10/13/87
3900 C* 10/13/87
4000 C*****
4100 * 2 03/20/89
4200 C READ TAG 10/13/87
4300 C READ RCVPART 9950 2 3 10/13/87
4400 C MAJCOD CABGT'03' ERROR 10/13/87
4500 C *IN44 CABEQ'1' END DET RECV? 10/13/87
4600 C MOVE RECID2 DBSEQ 10/13/87
4700 C MAJMIN CABEQ'0000' XMIT RCVTRNRND? 10/13/87
4800 C *IN40 CABEQ'1' XMIT RCVTRNRND? 10/13/87
4900 * 3 03/20/89
5000 C *IN40 DOWEQ'0' RCVTRNRND? B001 03/20/89
5100 C READ RCVTRND 9950 2 3 001 10/13/87
5200 C MAJCOD CABGT'03' ERROR 001 10/13/87
5300 C *IN44 CABEQ'1' END DETACH RECV? 001 10/13/87
5400 C END E001 10/13/87
5500 C*****
5600 C* 10/13/87
5700 C* A REQUEST FROM THE SOURCE PROGRAM RESULTS IN READING A SINGLE 10/13/87
5800 C* RECORD CONTAINING THE REQUESTED CUSTOMER OR ORDER NUMBER. THE 10/13/87
5900 C* RESPONSE WILL BE RETURNED IN A SINGLE RECORD CONTAINING EITHER 10/13/87
6000 C* THE ITEM OR CUSTOMER INFORMATION, DEPENDING ON THE DATABASE 10/13/87
6100 C* CONTENT. 10/13/87
6200 C* 10/13/87
6300 C* THE RESPONSE IS SENT TO THE SOURCE PROGRAM BY WRITING TO THE 10/13/87
6400 C* ICF FILE USING FORMAT SNDPART. 10/03/90
6500 C* 10/13/87
6600 C*****
6700 * 4 03/20/89
6800 C XMIT TAG 10/13/87
6900 C DBSEQ CHAINPFILE 98 98 IF NOT FD 1 03/20/89
7000 C MOVE DBSEQ ITEMNO 10/13/87
7100 C MOVE RECCUS RECTYP RECD FMT ID 10/13/87
7200 C*****
7300 C* 03/20/89
7400 C* WHEN THE REQUESTED CUSTOMER OR ITEM NUMBER IS NOT FOUND, 10/13/87
7500 C* 000000 IS PROPAGATED TO THE KEY FIELD BEFORE THE RESPONSE 10/13/87
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTDMUL 10/03/90 14:53:44 Page 4
SEQUENCE
NUMBER *...1....+...2....+...3....+...4....+...5....+...6....+...7...* USE NUM LAST PAGE PROGRAM
                UPDATE LINE ID
7600 C* IS SENT BACK TO THE SOURCE PROGRAM. 10/13/87
7700 C* 03/20/89
7800 C*****
7900 C 98 MOVE '000000' ITEMNO 10/13/87
8000 C MOVE LDBDATA EDATA MOVE DATA 10/13/87
8100 C WRITESNDPART DATA W/DET 10/13/87
8200 C MAJCOD CABGT'03' ERROR 10/13/87
8300 C GOTO READ 10/13/87
8400 C*****
8500 C* 10/13/87
8600 C* IF ANY ICF FILE ERROR OCCURS, PRINT THE ERROR MESSAGE, 10/03/90
8700 C* AND THEN END THE JOB. 10/13/87
8800 C* 10/13/87
8900 C*****

```

Figure 11-24 (Part 2 of 4). Target Program Example — RTDMUL (User-Defined Formats)

```

9000 * 5 03/20/89
9100 C ERROR TAG 10/13/87
9200 C EXCPTMERR 10/13/87
9300 C END TAG 10/13/87
9400 * 6 03/20/89
9500 C SETON LR 1 10/13/87
9600 C RETRN 10/13/87
9700 C***** 10/14/87
9800 C* 10/14/87
9900 C* THIS IS THE PROGRAM EXCEPTION/ERROR SUBROUTINE THAT RECEIVES
10000 C* CONTROL WHEN AN EXCEPTION OR ERROR OCCURS AFTER AN I/O IS 03/20/89
10100 C* ISSUED TO AN ICF PROGRAM DEVICE AND THERE IS A NON-ZERO 03/20/89
10200 C* VALUE UPDATED IN THE RPG STATUS FIELD (ERR). THIS ROUTINE 03/20/89
10300 C* CHECKS FOR STATUS VALUES THAT RELATE TO AN ICF OPERATION. 03/20/89
10400 C* IF THE PROGRAM DEVICE IS ALREADY ACQUIRED, THE EXCEPTION IS 03/20/89
10500 C* IGNORED, OTHERWISE THE PROGRAM IS TERMINATED. 03/20/89
10600 C* 10/14/87
10700 C***** 10/14/87
10800 * 7 03/20/89
10900 C *PSSR BEGSR 10/14/87
11000 C MOVE ' ' RETURN 6 DEFAULT 10/14/87
11100 C ERR CABEQ01285 ENDPSR ALREADY ACQ? 10/14/87
11200 C MOVE '*CANCL' RETURN JOB ENDS 10/14/87
11300 C ENDPSR ENDSRRETURN BACK TO MAIN 10/14/87
11400 C***** 10/13/87
11500 QPRINT E 1 MMERR 10/13/87
11600 O 21 'ERROR ON ICF ' 10/03/90
11700 O 34 'MAJOR/MINOR:' 10/13/87
11800 O MAJCOD 37 10/13/87
11900 O 38 '/' 10/13/87
12000 O MINCOD 40 10/13/87
12100 O 49 'FORMAT:' 10/13/87
12200 O FMTNM 60 10/13/87
12300 O 69 'PGMDEV:' 10/13/87
12400 O PGMDEV 80 10/13/87
* 6103 12401 OVERFLOW INDICATOR 0A ASSIGNED TO FILE QPRINT.
E000000 OUTPUT FIELDS FOR RECORD SNDPART FILE CFIL FORMAT SNDPART.
E000001 RECTYP 1 CHAR 1
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RTDMUL 10/03/90 14:53:44 Page 5
SEQUENCE IND DO LAST PAGE PROGRAM
NUMBER *...1....2....3....4....5....6....7...* USE NUM UPDATE LINE ID
E000002 ITEMNO 7 CHAR 6
E000003 EDATA 137 CHAR 130
E000004 FILL1 150 CHAR 13
***** END OF SOURCE *****
Additional Diagnostic Messages
* 7089 1300 RPG PROVIDES SEPARATE INDICATOR AREA FOR FILE CFIL.
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RTDMUL 10/03/90 14:53:44 Page 6
Key Field Information
PHYSICAL LOGICAL
FILE/RCD FIELD FIELD ATTRIBUTES
02 PFILE
DBREC
DBSEQ CHAR 6
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RTDMUL 10/03/90 14:53:44 Page 7
Cross Reference
File and Record References:
FILE/RCD DEV/RCD REFERENCES (D=DEFINED)
01 CFILE WORKSTN 1300D
RCVPART 1300D B000000 4300
RCVTRND 1300D C000000 5100
SNDPART 1300D A000000 8100 E000000
02 PFILE DISK 1600D 6900
DBREC 1600D D000000
03 QPRINT PRINTER 1700D 11500 12401
Field References:
FIELD ATTR REFERENCES (M=MODIFIED D=DEFINED)
*IN40 A(1) 4800 5000
*IN44 A(1) 4500 5300
*PSSR BEGSR 1300 10900D
DBDATA A(130) D000003D 8000
* 7031 DBFILL A(13) D000004D
DBSEQ A(6) D000002D 4600M 6900 7000
EDATA A(130) A000003D 8000M E000003D
END TAG 4500 5300 9300D
ENDPSR ENDSR 11100 11300D
ERR Z(5,0) 2100D 11100
ERROR TAG 4400 5200 8200 9100D
FEEDBK DS(404) 1300 1900D
FILL1 A(13) A000004D E000004D

```

Figure 11-24 (Part 3 of 4). Target Program Example — RTDMUL (User-Defined Formats)

```

* 7031 FILL4      A(64)  B000003D
      FMTNM      A(8)    2200D   12200
      ITEMNO     A(6)    A000002D  7000M   7900M E000002D
* 7031 LOC       A(8)    2000D
      MAJCOD     A(2)    2500D   4400   5200   8200   11800
      MAJMIN     A(4)    2400D   4700
      MINCOD     A(2)    2600D   1200D
      MMERR      EXCPT   9200   11500
* 7031 PARTDS   A(80)   B000002D
      PGMDEV     A(10)   2300D   12400
      READ       TAG     4200D   8300
      RECCUS     A(1)    D000001D  7100
      RECID2     A(6)    B000001D  4600
      RECTYP     A(1)    A000001D  7100M E000001D
      RETURN     A(6)    11000D  11200M  11300
      XMIT       TAG     4700   4800   6800D
      '          LITERAL  11000
      '*CANCL'   LITERAL  11200
      '0'        LITERAL  5000
      '0000'     LITERAL  4700
5738R61 V2R1M0  910524      IBM AS/400 RPG/400      ICFLIB/RTDMUL      10/03/90  14:53:44      Page      8
      '000000'   LITERAL  7900
      '03'       LITERAL  4400   5200   8200
      '1'        LITERAL  4500   4800   5300
      01285     LITERAL  11100
Indicator References:
INDICATOR REFERENCES (M=MODIFIED D=DEFINED)
*IN          4500   4800   5000   5300
LR           9500M
OA           1700D  12401
* 7031 05
* 7031 10
* 7031 15
* 7031 20
* 7031 25
* 7031 30
      40          4800   5000
      44          4500   5300
* 7031 50
      98          4300M  5100M
      99          6900M  7900
* 7031 99
      4300M  5100M
* * * * * E N D   O F   C R O S S   R E F E R E N C E   * * * * *
5738R61 V2R1M0  910524      IBM AS/400 RPG/400      ICFLIB/RTDMUL      10/03/90  14:53:44      Page      9
      M e s s a g e   S u m m a r y
* QRG6103 Severity: 00 Number: 1
      Message . . . . : No Overflow Indicator is specified but an
      indicator is assigned to a file and automatic skip to 6 is
      generated.
* QRG7031 Severity: 00 Number: 12
      Message . . . . : The Name or indicator is not referenced.
* QRG7089 Severity: 00 Number: 1
      Message . . . . : The RPG provides Separate-Indicator area for
      file.
* * * * * E N D   O F   M E S S A G E   S U M M A R Y   * * * * *
5738R61 V2R1M0  910524      IBM AS/400 RPG/400      ICFLIB/RTDMUL      10/03/90  14:53:44      Page      10
      F i n a l   S u m m a r y
Message Count: (by Severity Number)
      TOTAL  00   10   20   30   40   50
      14   14   0   0   0   0
Program Source Totals:
Records . . . . . : 124
Specifications . . . . . : 54
Table Records . . . . . : 0
Comments . . . . . : 70
PRM has been called.
Program RTDMUL is placed in library ICFLIB. 00 highest Error-Severity-Code.
* * * * * E N D   O F   C O M P I L A T I O N   * * * * *

```

Figure 11-24 (Part 4 of 4). Target Program Example — RTDMUL (User-Defined Formats)

```

5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFMUL          10/03/90 14:55:22      Page      1
Compiler . . . . . : IBM AS/400 RPG/400
Command Options:
  Program . . . . . : ICFLIB/RTFMUL
  Source file . . . . : ICFLIB/QICFPUB
  Source member . . . . : *PGM
Text not available for message RXT0073 file QRPMSG.
  Generation options . . . . . : *NOLIST *NOXREF *NOATR *NODUMP *NOOPTIMIZE
  Source listing indentation . . . . : *NONE
  SAA flagging . . . . . : *NOFLAG
  Generation severity level . . . . . : 9
  Print file . . . . . : *LIBL/QSYSPRT
  Replace program . . . . . : *YES
  Target release . . . . . : *CURRENT
  User profile . . . . . : *USER
  Authority . . . . . : *LIBCRTAUT
  Text . . . . . : *SRCMBRTXT
  Phase trace . . . . . : *NO
  Intermediate text dump . . . . . : *NONE
  Snap dump . . . . . : *NONE
  Codelist . . . . . : *NONE
  Ignore decimal data error . . . . . : *NO

```

```

Actual Program Source:
Member . . . . . : RTFMUL
File . . . . . : QICFPUB
Library . . . . . : ICFLIB
Last Change . . . . . : 10/03/90 14:52:20
Description . . . . . : RPG Multi-Session example w/FORMAT (target)
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFMUL          10/03/90 14:55:22      Page      2

```

SEQUENCE NUMBER	IND	DO	LAST UPDATE	PAGE LINE	PROGRAM ID
100	H*		10/13/87		
200	H*		03/20/89		
300	H*		10/13/87		
400	H*		10/13/87		
500	H*		10/13/87		
600	H*		03/20/89		
700	H*		03/20/89		
800	H*		03/20/89		
900	H*		10/13/87		
1000	H*		10/13/87		
1100	H*		03/20/89		
1200	H*		10/13/87		
	H				*****
1300	FCFILE	CF F	256		WORKSTN
1400	F				KINFDS FEEDBK
1500	F				KINFSR *PSSR
1600	FPFILE	IF E			K DISK
	RECORD FORMAT(S): LIBRARY ICFLIB FILE PFIL.				
	EXTERNAL FORMAT DBREC RPG NAME DBREC				
1700	FQPRINT	O F	132		PRINTER
1800	*1				38 45 FMTNM
1900	ICFILE	NS	99		
2000	I				1 6 RECID2
2100	I				7 150 PARTDS
A000000	INPUT				FIELDS FOR RECORD DBREC FILE PFIL
A000001					1 1 RECCUS
A000002					2 7 DBSEQ
A000003					8 137 DBDATA
A000004					138 150 DBFILL
2200	IFEEDBK		DS		
2300	I				*ROUTINE LOC
2400	I				*STATUS ERR
2500	I				38 45 FMTNM
2600	I				273 282 PGMDEV
2700	I				401 404 MAJMIN
2800	I				401 402 MAJCOD
2900	I				403 404 MINCOD
3000	C*				
3100	C*				
3200	C*				READ THE REQUEST FROM THE SOURCE PROGRAM. INDICATOR 40
3300	C*				INDICATES RCVTRNRND OCCURRED. INDICATOR 44 INDICATES THAT
3400	C*				DETACH HAS BEEN RECEIVED.
3500	C*				
3600	C*				INDICATOR 99 WILL BE TURNED ON FOR "I/O ERRORS" THEREBY

Figure 11-25 (Part 1 of 4). Target Program Example — RTFMUL (System-Supplied Formats)

```

5738R61 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFMUL          10/03/90 14:55:22          Page      3
SEQUENCE          IND      DO      LAST      PAGE      PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE      NUM      UPDATE      LINE      ID
3700 C* PREVENTING THE RPG DEFAULT ERROR HANDLER FROM BEING CALLED.          10/13/87
3800 C* THIS IS NECESSARY TO ALLOW THE PROGRAM TO PROCESS THE ICF          10/03/90
3900 C* MAJOR/MINOR RETURN CODE. THIS PROGRAM CHECKS FOR ERRORS ON          10/13/87
4000 C* EVERY ICF FILE OPERATION. A MAJOR CODE GREATER THAN 03          10/03/90
4100 C* INDICATES AN ERROR.          10/13/87
4200 C*          10/13/87
4300 C*****          10/13/87
4400 C* 2          03/20/89
4500 C          READ          TAG          10/13/87
4600 C          READ CFIL          9950          2 3          10/13/87
4700 C          MAJCOD          CABGT'03'          ERROR          SESSION ERR          10/13/87
4800 C          MINCOD          CABEQ'08'          END          DETACH RECV?          10/13/87
4900 C          MOVE RECID2          DBSEQ          SAVE RECD #          10/13/87
5000 C          MAJMIN          CABEQ'0000'          XMIT          RCVTRNRND?          10/13/87
5100 C          MAJMIN          CABEQ'0300'          XMIT          RCVTRNRND?          10/13/87
5200 C* 3          03/20/89
5300 C          MINCOD          DOWNE'00'          RCVTRNRND?          B001          10/13/87
5400 C          READ CFIL          9950          2 3          001          10/13/87
5500 C          MAJCOD          CABGT'03'          ERROR          001          10/13/87
5600 C          MINCOD          CABEQ'08'          END          DETACH RECV?          001          10/13/87
5700 C          END          E001          10/13/87
5800 C*****          10/13/87
5900 C*          03/20/89
6000 C* A REQUEST FROM THE SOURCE PROGRAM RESULTS IN READING A SINGLE          10/13/87
6100 C* RECORD CONTAINING THE REQUESTED CUSTOMER OR ORDER NUMBER. THE          10/13/87
6200 C* RESPONSE WILL BE RETURNED IN A SINGLE RECORD CONTAINING EITHER          10/13/87
6300 C* THE ITEM OR CUSTOMER INFORMATION, DEPENDING ON THE DATABASE          10/13/87
6400 C* CONTENT.          10/13/87
6500 C*          10/13/87
6600 C* THE RESPONSE IS SENT TO THE SOURCE PROGRAM BY WRITING TO THE          10/13/87
6700 C* ICF FILE USING FORMAT SNDPART.          10/03/90
6800 C*          03/20/89
6900 C*****          10/13/87
7000 C* 4          03/20/89
7100 C          XMIT          TAG          10/13/87
7200 C          DBSEQ          CHAINPFILE          98          98 IF NOT FD          1          10/13/87
7300 C          MOVE DBSEQ          RECID2          10/13/87
7400 C*****          10/13/87
7500 C*          03/20/89
7600 C* WHEN THE REQUESTED CUSTOMER OR ITEM NUMBER IS NOT FOUND,          10/13/87
7700 C* 000000 IS PROPAGATED TO THE KEY FIELD BEFORE THE RESPONSE          10/13/87
7800 C* IS SENT BACK TO THE SOURCE PROGRAM.          10/13/87
7900 C*          03/20/89
8000 C*****          10/13/87
8100 C          98          MOVE '000000'          RECID2          10/13/87
8200 C          EXCPTSNDITM          DATA          10/13/87
8300 C          MAJCOD          CABGT'03'          ERROR          10/13/87
8400 C          GOTO READ          10/13/87
8500 C*****          10/13/87
8600 C*          10/13/87
8700 C* IF ANY ICF FILE ERROR OCCURS, PRINT THE ERROR MESSAGE,          10/03/90
8800 C* AND THEN END THE JOB.          10/13/87
8900 C*          10/13/87
9000 C*****          10/13/87

5738R61 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFMUL          10/03/90 14:55:22          Page      4
SEQUENCE          IND      DO      LAST      PAGE      PROGRAM
NUMBER *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE      NUM      UPDATE      LINE      ID
9100 C* 5          03/20/89
9200 C          ERROR          TAG          10/13/87
9300 C          EXCPTMMERR          10/13/87
9400 C          END          TAG          10/13/87
9500 C* 6          03/20/89
9600 C          SETON          LR          1          10/13/87
9700 C          RETRN          10/13/87
9800 C*****          10/14/87
9900 C*          10/14/87
10000 C* THIS IS THE PROGRAM EXCEPTION/ERROR SUBROUTINE THAT RECEIVES          10/14/87
10100 C* CONTROL WHEN AN EXCEPTION OR ERROR OCCURS AFTER AN I/O          10/14/87
10200 C* IS ISSUED TO AN ICF PROGRAM DEVICE AND THERE IS A          10/03/90
10300 C* NON-ZERO VALUE UPDATED INTO THE RPG STATUS FIELD (ERR).          01/19/88
10400 C* THIS ROUTINE CHECKS FOR STATUS VALUES THAT RELATE TO          01/19/88
10500 C* ICF OPERATION.          10/03/90
10600 C* IF THE PROGRAM DEVICE IS ALREADY ACQUIRED, THE EXCEPTION IS          03/20/89
10700 C* IGNORED, OTHERWISE THE PROGRAM IS TERMINATED.          03/20/89
10800 C*          10/14/87
10900 C*****          10/14/87

```

Figure 11-25 (Part 2 of 4). Target Program Example — RTFMUL (System-Supplied Formats)

```

11000 * 7 03/20/89
11100 C *PSSR BEGSR 10/14/87
11200 C MOVE ' ' RETURN 6 DEFAULT 10/14/87
11300 C ERR CABEQ01285 ENDPSR ALREADY ACQ? 10/14/87
11400 C MOVE '*CANCL' RETURN JOB ENDS 10/14/87
11500 C ENDPSR ENDSRRETURN BACK TO MAIN 03/20/89
11600 OQPRINT E 1 MMERR 10/13/87
11700 0 21 'ERROR ON ICF ' 10/03/90
11800 0 34 'MAJOR/MINOR:' 10/13/87
11900 0 MAJCOD 37 10/13/87
12000 0 38 '/' 10/13/87
12100 0 MINCOD 40 10/13/87
12200 0 49 'FORMAT:' 10/13/87
12300 0 FMTNM 60 10/13/87
12400 0 69 'PGMDEV:' 10/13/87
12500 0 PGMDEV 80 10/13/87
12600 OCFILE E SNDITM 10/13/87
12700 0 K6 '$$SEND' 10/13/87
12800 0 4 '0150' 10/13/87
12900 0 RECCUS 5 10/13/87
13000 0 RECID2 11 10/13/87
13100 0 DBDATA 141 10/13/87
13200 0 DBFILL 154 10/13/87
* 6103 13201 OVERFLOW INDICATOR OA ASSIGNED TO FILE QPRINT.
***** END OF SOURCE *****
Additional Diagnostic Messages
* 7089 1300 RPG PROVIDES SEPARATE INDICATOR AREA FOR FILE CFIL.
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RTFMUL 10/03/90 14:55:22 Page 5
Key Field Information
PHYSICAL LOGICAL
FILE/RCD FIELD FIELD ATTRIBUTES
02 PFILE
DBREC
DBSEQ CHAR 6
5738RG1 V2R1M0 910524 IBM AS/400 RPG/400 ICFLIB/RTFMUL 10/03/90 14:55:22 Page 6
Cross Reference
File and Record References:
FILE/RCD DEV/RCD REFERENCES (D=DEFINED)
01 CFILE WORKSTN 1300D 1900 4600 5400 12600
$$SEND 12700
02 PFILE DISK 1600D 7200
DBREC 1600D A000000
03 QPRINT PRINTER 1700D 11600 13201
Field References:
FIELD ATTR REFERENCES (M=MODIFIED D=DEFINED)
*PSSR BEGSR 1300 11100D
DBDATA A(130) A000003D 13100
DBFILL A(13) A000004D 13200
DBSEQ A(6) A000002D 4900M 7200 7300
END TAG 4800 5600 9400D
ENDPSR ENDSR 11300 11500D
ERR Z(5,0) 2400D 11300
ERROR TAG 4700 5500 8300 9200D
FEEDBK DS(404) 1300 2200D
FMTNM A(8) 2500D 12300
* 7031 LOC A(8) 2300D
MAJCOD A(2) 2800D 4700 5500 8300 11900
MAJMIN A(4) 2700D 5000 5100
MINCOD A(2) 2900D 4800 5300 5600 12100
MMERR EXCPT 9300 11600
* 7031 PARTDS A(144) 2100D
PGMDEV A(10) 2600D 12500
READ TAG 4500D 8400
RECCUS A(1) A000001D 12900
RECID2 A(6) 2000D 4900 7300M 8100M 13000
RETURN A(6) 11200D 11400M 11500
SNDITM EXCPT 8200 12600
XMIT TAG 5000 5100 7100D
' LITERAL 11200
'*CANCL' LITERAL 11400
'00' LITERAL 5300
'0000' LITERAL 5600
'000000' LITERAL 8100
'03' LITERAL 4700 5500 8300
'0300' LITERAL 5100
'08' LITERAL 4800 5600
01285 LITERAL 11300

```

Figure 11-25 (Part 3 of 4). Target Program Example — RTFMUL (System-Supplied Formats)

```

5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFMUL          10/03/90 14:55:22          Page 7
Indicator References:
INDICATOR REFERENCES (M=MODIFIED D=DEFINED)
LR          9600M
OA          1700D 13201
* 7031 50   4600M 5400M
          98   7200M 8100
* 7031 99   1900M 4600M 5400M
***** END OF CROSS REFERENCE *****
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFMUL          10/03/90 14:55:22          Page 8
                                Message Summary
* QR66103 Severity: 00 Number: 1
  Message . . . . : No Overflow Indicator is specified but an
  indicator is assigned to a file and automatic skip to 6 is
  generated.
* QR67031 Severity: 00 Number: 4
  Message . . . . : The Name or indicator is not referenced.
* QR67089 Severity: 00 Number: 1
  Message . . . . : The RPG provides Separate-Indicator area for
  file.
***** END OF MESSAGE SUMMARY *****
5738RG1 V2R1M0 910524          IBM AS/400 RPG/400          ICFLIB/RTFMUL          10/03/90 14:55:22          Page 9
                                Final Summary
Message Count: (by Severity Number)
TOTAL  00  10  20  30  40  50
       6   6   0   0   0   0
Program Source Totals:
Records . . . . . : 132
Specifications . . . . . : 62
Table Records . . . . . : 0
Comments . . . . . : 70
PRM has been called.
Program RTFMUL is placed in library ICFLIB. 00 highest Error-Severity-Code.
***** END OF COMPILATION *****

```

Figure 11-25 (Part 4 of 4). Target Program Example — RTFMUL (System-Supplied Formats)

Chapter 12. Tracing Intersystem Communications Function Operations and Functions

You can use the Trace Intersystem Communications Function (TRCICF) command to save information about the language operations and communications functions directed to an ICF file. The trace information can be collected in the current job or in the job being serviced as a result of the Start Service Job (STRSRVJOB) command.

The Start Service Job (STRSRVJOB) command allows you to collect trace records for jobs started from other work stations or for batch jobs. After the STRSRVJOB command has been entered, the TRCICF command must be entered to start the trace.

The End Service Job (ENDSRVJOB) command is used to end the service job request. No parameters are used with this command. The trace must be stopped before this command can be entered. The *CL Reference* manual has more information about the STRSRVJOB and ENDSRVJOB commands.

Trace ICF traces all ICF I/O operations that occur in the job in which the command was entered. During the time that TRCICF is active, all programs that run in the job are monitored by TRCICF. TRCICF can be entered within different jobs, and the trace for one job runs independently of the trace for another job.

The Trace ICF function can be started, stopped, or ended. You can start the Trace ICF function from a system menu, by typing the TRCICF command on a command line, or from a control language (CL) program within a job. After the trace is started, trace records are collected and stored in an internal trace storage area. When the trace is stopped, the trace records can either be directed to the spooled printer file, QPIFTRCF, or sent to a database output file that you specify. When the trace is ended, all trace records are deleted. Details about starting, stopping, and ending the Trace ICF function are discussed in this chapter.

Starting the Trace

The Trace ICF (TRCICF) function can be started before running a job or after the job is active (as in a remote job). You can start TRCICF from a system menu, by typing TRCICF *ON on any command line, by adding the command to a CL program, or by typing TRCICF on a command line and pressing F4 (Prompt). If the latter method is used, an initial prompt is displayed for the *Trace option setting*. If *ON is specified and you press the Enter key, the following display is shown:

```
TRACE ICF (TRCICF)

Type choices, press Enter.

Trace option setting . . . . . *ON          *ON, *OFF, *END
Maximum storage to use . . . . . 288        1-16000 K
Trace full . . . . . *WRAP                *WRAP, *STOPTRC
User data length . . . . . 128             0-4096

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

Trace option setting

Specify whether collecting trace information is to be started, stopped, or ended.

*ON

Trace ICF is to be started. This is the default value for this prompt.

*OFF

Trace ICF is stopped. No other trace information is recorded and the current information is written to the spooled printer file or a database file.

*END

Trace ICF is ended. All trace information is deleted.

Maximum storage to use

Specify the maximum amount of storage to use for the trace information collected. This prompt is only shown if you have selected *ON for the *Trace option setting* prompt.

200KB

The number of bytes (1KB equals 1024 bytes) of maximum storage. This is the default value.

1-16000KB

The valid range for the number of bytes of maximum storage.

Trace full

Specify whether new trace records are to replace the old trace records or to stop the trace function when the maximum storage specified has been reached. This prompt is only shown if you have selected *ON for the *Trace option setting* prompt. Valid values are:

*WRAP

When the trace storage area is full, new trace information is written over the oldest information, starting at the beginning of the storage area. This is the default value.

*STOPTRC

No new trace information is saved when the trace storage area is full. You must turn the trace off to get the output.

User data length

Specify the maximum length of user data to be saved for each trace record in the storage area. This prompt is only shown if you have selected *ON for the *Trace option setting* prompt.

128

The number of bytes for the user data length. This is the default value.

0-4096

The valid range of bytes for the user data length.

Stopping the Trace

Trace ICF continues to collect trace records until you stop the trace, or until the trace storage area is full. When you stop the trace, the trace records that have been created are either directed to the spooled printer file, QPIFTRCF, or to a database output file that you specify. If the output file specified already exists, it must have the same attributes as the system-supplied file QAIFTRCF.

You can stop the trace from a system menu, by typing TRCICF *OFF on any command line, by adding the command to a CL program, or by typing TRCICF and pressing F4 (prompt). If the latter method is used, and you specify *OFF for the *Trace option setting*, you are prompted for the OUTPUT parameter. If you specify the *OUTFILE option for the *Output* prompt, the following display is shown:

```
TRACE ICF (TRCICF)
Type choices, press Enter.
Trace option setting . . . . . > *OFF      *ON, *OFF, *END
Output . . . . . > *OUTFILE      *PRINT, *OUTFILE
Output file . . . . . _____ Name
Library . . . . . _____ *LIBL  Name, *LIBL, *CURLIB
Output member options:
Member to receive output . . . *FIRST  Name, *FIRST
Replace or add records . . . *REPLACE *REPLACE, *ADD
```

Output

Specifies whether the trace information is to be stored in a spooled file or saved in a database file. This display is only shown if *OFF is specified for the *Trace option setting* prompt. Valid values are:

*PRINT

The trace information is sent to the spooled file QPIFTRCF in the output queue associated with the job being traced. The spooled file can be viewed or printed. Refer to Figure 12-1 on page 12-4 for an example of the spooled trace records. This is the default value.

*OUTFILE

The trace records are to be directed to a database file. Refer to "Trace Records Sent to a Database File" on page 12-5 for a description of trace records directed to a database file. The

*OUTFILE value for the *Output* prompt is only valid if a value is specified for the *Output file* prompt.

Output file

Specifies the name of the database file to which the trace records are to be sent. This prompt is only shown if you have selected *OFF for the *Trace option setting* prompt and *OUTFILE for the *Output* prompt. If the file does not exist, the system creates a new database file with the specified name in the library to which the file is to be added. The new file has the same attributes as the system-supplied file QAIPTRCF. If the file already exists, it must have the same attributes as the system-supplied file QAIPTRCF. Possible library values are:

Name

The name of the library where the file is located.

*LIBL

The file is located in the library list.

*CURLIB

The file is located in the current library for the job. If no current library entry exists in the library list, the library QGPL is used.

Output member options

Specifies the name of the file member that is to receive the trace information. This prompt is only shown if you have selected *OFF for the *Trace option setting* prompt and *OUTFILE for the *Output* prompt. If the output file is to be created by the system, an output member is also created and given the name specified in the *Member to receive output* prompt. If *FIRST is specified for the *Member to receive output* prompt, a member is created and given the name specified in the output file. If the output file exists, but the output member does

not, a member with the specified name is created. The options for the *Output member options* prompt are:

Member to receive output

Type the name of the member to receive the output. Valid values are:

*FIRST

The first member in the output file receives the trace information. This is the default.

Name

The specified member receives the trace information.

Replace or add records

The trace information is either added to the file or replaces existing trace information. Valid values are:

*REPLACE

New trace information replaces what is already in the file member. This is the default.

*ADD

New trace information is appended to the end of data already in the file member.

Trace Records Sent to a Spooled File

When you select *OFF for the *Trace option setting* prompt and press F4, you are presented with the option on the *Output* prompt to send the trace records to a spooled file (*PRINT) or to a database file (*OUTFILE). The default value is *PRINT. If you choose the *PRINT value for the *Output* prompt, the trace information is sent to the spooled file QPIPTRCF. Figure 12-1 on page 12-4 shows the format of the spooled trace records.

```

-----Table of Function Codes-----
1 Function Codes      Meaning
ACQ                   Acquire
AWT                   Allow-Write
CFM                   Confirm
CLS                   Close File Prior to REL or EOS
CNI                   Cancel-Invite
CNL                   Cancel
DET                   Detach
EGP                   End-of-Group
EOA                   End-of-Session-Abnormal
EOS                   End-of-Session
ERR                   Error: Function Not Valid
EVK                   Evoke
FAL                   Fall
FMH                   Function-Management-Header
FMT                   Format-Name
FRC                   Force-Data
GTA                   Get-Attributes
INV                   Invite
NRP                   Negative-Response
CPN                   Open with Acquire-Program-Device
RCF                   Respond-to-Confirm
RCV                   Receive
REL                   Release
RFI                   Read-From-Invited-Program-Devices
RST                   Restore
RWT                   Request-to-Write
SDV                   Subdevice-Selection
SND                   Send
SPD                   Suspend
TMR                   Timer
TRN                   Turn-Around

```

```

5738SS1 V2R1M0 910524      AS/400 Trace ICF Information      10/02/90 13:18:14      PAGE 2
2 Job . . . . : DSP10      3 User . . . . : QUSER      4 Number . . . . : 00626
5 Program . . . . : ICFTEST /ICFMAIN      6 Program File . . . . : ICFTEST /ICFTSTCF
7 Opened File . . . . : ICFLIB /ICFTSTITF
8 Program      9 Record      10 Return      11      12      13 Data      14 Remote      15
Device         Format         Code         Function      Response      Length      Location      Time
Indicator
DEV1           EVOKENOV      0000         ACQ           0             0           Chicago      15:37:00.857
DEV1           NOVARLEN      0001         SND,INV       0             0           Chicago      15:37:06.264
DEV1           NOVARLEN      0001         SND,INV       B0            0           Chicago      15:37:38.798
DATA:
THIS IS A PUT WITH INVITE.
DEV1           NOVARLEN      0308         RCV           DET           0           Chicago      15:37:50.097
DEV1           EVOKENOV      0000         EVK           0             0           Chicago      15:38:06.887
DEV1           NOVARLEN      0001         SND,INV       B0            0           Chicago      15:38:38.912
DATA: 16
THIS IS ANOTHER PUT WITH INVITE.
DEV1           NOVARLEN      0308         RCV           DET           0           Chicago      15:38:52.835
DEV1           NOVARLEN      0000         EOS           0             0           Chicago      15:40:43.560

```

***** END OF LISTING *****

Figure 12-1. Spooled Trace Records

1 Table of Function Codes

The first page of the spooled trace records is a table of the function codes used for each ICF operation. The function code is printed in the Function and Response Indicator columns.

Notes:

1. The suspend (SPD) and restore (RST) function codes are used in the System/36 environment. These function codes are part of the read-under-format (RUF) support. If two programs using the RUF support do not run in the same job, then the Trace ICF function does not trace

both programs unless Trace ICF is started for both jobs.

2. The function code OPN indicates that the program opened a file that automatically acquired a program device. CLS indicates that the program closed the file prior to releasing or ending the session.

2 Job

Name of the job in which your program is running.

3 User

The user identification (User ID) used to start the job (either the user ID used to

sign on the work station or the user ID received on the program start request).

4 Number

The number assigned to the job step when your program started.

5 Program

Name of the library where the program resides, and the name of the program that issued the operation that is being traced.

6 Program File

Name of the library where the ICF file named in the program resides, and the name of the ICF file named in the program.

7 Opened File

Name of the library where the ICF file opened by your program resides, and the name of the ICF file opened by your program.

Note: If you used the OVRICFF command to temporarily override the file named in the program, the name specified for the Opened File will be different from the Program File name.

8 Program Device

Name assigned to the session to which the language operation or communications function was directed. This is the name specified in the Add ICF Program Device Entry (ADDICFDEVE) or Override ICF Program Device Entry (OVRICFDEVE) commands.

9 Record Format

Name of the record format used when the communications function is issued. The record format can either be a user-defined data description specification (DDS) or a system-supplied format.

10 Return Code

The major and minor return code issued to indicate the success or failure of each operation.

11 Function

The function code assigned to represent the language operation or communications function issued by the program. Only operations associated with your ICF sessions are traced. File open and close operations are not traced except when a

program device is acquired or released as a result of an open or close operation.

12 Response Indicator

The function code assigned to represent the DDS response indicator that indicates status information about the input operation.

13 Data Length

Length of data sent or received by the program. If the function indicates a send and receive operation, then this field represents the length of data received by the program.

14 Remote Location

Name of the remote location with which a communication session is established.

15 Time

Time that the language operation or communication function was completed by the communications type. The time is displayed in hours, minutes, seconds, and milliseconds.

16 Data

The data sent or received by the program. The amount of data traced depends on the value specified for the *User data length* prompt (DTALEN parameter) of the TRCICF command. If the function indicates a send and receive operation, then the data received by your program is shown.

Trace Records Sent to a Database File

When you select *OFF for the *Trace option setting*, you are presented with the option either to send the trace records to a spooled file (*PRINT) or send the records to a database file (*OUTFILE). If you choose the *OUTFILE value for the *Output* prompt, the trace information is sent to the database file that you specify. If you specify a file that already exists, it must match the attributes of the system-supplied file QAIFTRCF.

The following example shows the layout of the trace records sent to a database file. The database file has a fixed record length of 4337 decimal bytes. The record format name is QIFTRC. Each record in the file contains all the information related to the language operation or

communications function, as well as the length of data traced. The length of data traced is less than or equal to the value specified on the *User data length* prompt (the DTALEN parameter) of the TRCICF command. Database files contain much of the same information that is contained in spooled files. Database files also contain the century and system name. For the 20th century, CENTURY would be 0 and for the 21st century, CENTURY would be 1, and so on. Date is in the YYMMDD format.

```
A*
A* TRACE ICF OUTFILE RECORD FORMAT FOR TRCICF
A*
A  R QIFTRC      TEXT('TRCICF Record')
A  IFJOB       10 COLHDG('JOB' 'NAME')
A              TEXT('Name of Job')
A  IFUSER       10 COLHDG('USER' 'NAME')
A              TEXT('Name of User')
A  IFNBR        6  COLHDG('JOB' 'NUMBER')
A              TEXT('Number of Job')
A  IFPGM        10 COLHDG('PROGRAM' 'NAME')
A              TEXT('Name of Program')
A  IFLIB        10 COLHDG('LIBRARY' 'NAME')
A              TEXT('Programs Library')
A  IFPGMF       10 COLHDG('PROGRAM' 'FILE' 'NAME')
A              TEXT('Program File')
A  IFPGML       10 COLHDG('PROGRAM' 'FILE' 'LIBRARY')
A              TEXT('Program Files Library')
A  IFOPNF       10 COLHDG('OPENED' 'FILE' 'NAME')
A              TEXT('Opened File')
A  IFOPNL       10 COLHDG('OPENED' 'FILE' 'LIBRARY')
A              TEXT('Opened Files Library')
A  IFPGDV       10 COLHDG('PROGRAM' 'DEVICE')
A              TEXT('Program Device')
A  IFRCFM       10 COLHDG('RECORD' 'FORMAT')
A              TEXT('Record Format')
A  IFMJMN       4  COLHDG('RETURN' 'CODE')
A              TEXT('Return Code')
A  IFOPCD       48 COLHDG('FUNCTION' 'CODE')
A              TEXT('Function Code')
A  IFRSPI       36 COLHDG('RESPONSE' 'INDICATOR')
A              TEXT('Response Indicator')
A  IFDTLN       2B COLHDG('DATA' 'LENGTH')
A              TEXT('Data Length')
A  IFRLOC       8  COLHDG('REMOTE' 'LOCATION')
A              TEXT('Remote Location')
A  IFTIME       9S 0 COLHDG('TIME')
A              TEXT('Time of Entry')
A  IFDTTR       2B COLHDG('TRACED' 'DATA' 'LENGTH')
A              TEXT('Traced Data Length')
A  IFCENT       1  COLHDG('CENTURY')
A              TEXT('Century of Entry')
A  IFDATE       6  COLHDG('DATE')
A              TEXT('Date of Entry')
A  IFSYS        8  COLHDG('SYSTEM' 'NAME')
A              TEXT('System Name')
A  IFRES        11 COLHDG('RESERVED')
A              TEXT('Reserved')
A  IFDATA      4096 COLHDG('DATA')
A              TEXT('Data')
```

Ending the Trace

You can end TRCICF from a system menu, by typing the TRCICF *END command on any command line, by adding the command to a CL program, or by typing TRCICF and pressing F4 to show the *Trace option setting* prompt, shown following. Type *END and press the Enter key. This causes Trace ICF to end and all trace records to be deleted.

```
TRACE ICF (TRCICF)

Type choices, press Enter.

Trace option setting . . . . . *OFF      *ON, *OFF, *END
```

Additional Considerations

Trace ICF traces only those operations that are associated with your ICF sessions. For example, ICF file open and close operations are not traced except when a program device is acquired or released as a result of an open or close operation. The following restrictions apply to Trace ICF traces:

- When an open of an ICF file is issued without implicit acquire of the program device, the explicit acquire of the program device (ACQ) will be traced and not the open (OPN) operation.
- An open of an ICF file with implicit acquire of the program device is traced as an open operation (OPN).
- When the close of an ICF file is preceded with an end of session (EOS), the end of session is traced but not the close (CLS).
- When the close of an ICF file is preceded by a release operation (REL), the release operation is traced but not the close operation (CLS).
- When a close of an ICF file is not preceded by an EOS or release operation, it is traced as a close (CLS) operation.
- When receiving program initialization parameters (sent by means of an evoke operation) in the System/36 environment with a read operation, the *Data length* field in the trace

| output will be 0, even if data was actually
| received by your program.

- When using BSCEL with data compression, the actual length of the data received on an input operation is not known by the system after decompression. The traced data length will be 0 and the traced data will not appear in the trace record, even if data was actually received by the program.

| **Displaying Communications Status**

| You can obtain current status information about
| operations and functions directed to an ICF file
| for all active (acquired) sessions within a job by
| using the Display Job (DSPJOB) or Work with
| Job (WRKJOB) commands. Choose option 17,
| Display communications status, from a Display
| Job or Work with Job display. You can also
| access this information from the Work with
| Active Jobs (WRKACTJOB) display. Refer to the
| *Communications Management Guide* for more
| information.

Appendix A. Language Operations, Data Description Specifications Keywords, and System-Supplied Formats

This appendix contains charts which show the following:

- All valid language operations supported by intersystem communications function (ICF)
- All valid operations for each programming language that supports ICF, C/400, COBOL/400, FORTRAN/400, and RPG/400 programming languages
- Data description specifications (DDS) processing keywords supported by communications types
- System-supplied formats supported by communications types

Language Operations

Figure A-1 describes the language operations supported by ICF.

Figure A-1. Language Operations

ICF Operations	Description
Open	Opens the ICF file.
Acquire	Establishes a session between the application and the remote location.
Get attributes	Used to determine the status of the session.
Read	Obtains data from a specific session.
Read-from-invited-program-devices	Obtains data from any session that has responded to an invite function.
Write	Passes data records from the issuing program to the other program in the transaction.
Write/Read	Allows a write operation followed by a read operation. Valid for C/400 and RPG/400 programming languages.
Release	Attempts to end a session.
Close	Closes the ICF file.

Figure A-2 shows all the valid communications operations for each programming language that

supports ICF (C/400, COBOL/400, RPG/400 programming languages).

Note: C/400 statements are case sensitive.

Figure A-2. Language Operations

ICF Operation	C/400 Function	COBOL/400 Procedure Statement	RPG/400 Operation	FORTTRAN/400 Statement
Open	fopen, _Ropen	OPEN	OPEN	OPEN
Acquire	QXXACQUIRE, _Racquire	ACQUIRE	ACQ	Not supported ²
Get-attributes	QXXDEVATR, _Rdevatr	ACCEPT	POST	Not supported
Read	fread, _Rreadn	READ	READ	READ
Read-from-invited-program-devices	QXXREADINVDEV followed by an fread, _Rreadindv	READ ¹	READ ¹	Not supported
Write	fwrite, _Rwrite	WRITE	WRITE	WRITE
Write/Read	_Rwriterd	Not supported	EXFMT	Not supported
Release	QXXRELEASE, _Rrelease	DROP	REL	Not supported
Close	fclose, _Rclose	CLOSE	CLOSE	CLOSE

¹ A read operation can be directed either to a specific program device or to any invited program device. The support provided by the compiler you are using determines whether to issue an ICF read or read-from-invited-program-devices operation, based on the format of the read operation. For example, if a read operation is issued with a format or terminal specified, the read operation is interpreted as an ICF read operation. Refer to the appropriate language reference manual for more information.

² To acquire a program device using the FORTRAN/400 language, you must specify the program device on the ACQPGMDEV parameter on the CRTICFF, CHGICFF, or OVRICFF commands. The program device will then be implicitly acquired when the ICF file is opened.

DDS Keyword Support

Figure A-3 defines the communications processing control DDS keywords supported for the ICF file, and the communications type that supports these keywords.

Figure A-3 (Page 1 of 4). Processing Control DDS Keywords

DDS Keyword	APPC	SNUF	BSCCL	Asyn-chronous	Intra-system	Finance	Retail
ALVWRT ² The record currently being written ends a transmission. The program goes to receive state.	X	X	X		X		
CANCEL Cancels a group of records that has just been sent.		X			X	X ³	X

Figure A-3 (Page 2 of 4). Processing Control DDS Keywords

DDS Keyword	APPC	SNUF	BSECEL	Asyn- chronous	Intra- system	Finance	Retail
CNLINVITE Cancels any valid invite issued by your program.		X	X	X	X	X	X
CONFIRM² Requests that the remote program confirm receiving data.	X				X		
DETACH Informs the remote program that the sending program is ending the transaction.	X	X	X	X ¹	X		X
ENDGRP Indicates the end of a user-defined group of records.		X	X		X	X	X
EOS Used to specify an end-of-session function.	X	X	X	X	X	X	X
EVOKE Starts a program on the remote system.	X	X	X	X	X		X
FAIL Sends a fail indication to the remote system.	X	X	X	X	X	X	
FMH Informs the remote program that a function-management-header (FMH) is being sent.		X		X ¹	X	X	X
FMTNAME² Specifies that the format name should be sent on output operations.	X				X		
FRCDTA² Immediately sends communications data currently in the buffer, without waiting for the buffer to become full.	X				X	X	X
INVITE Schedules an invite.	X	X	X	X	X	X	X
NEGRSP Informs the remote system that the data received is not valid.		X			X	X	X
RCVCANCEL² Indicates that the remote program sent a cancel request.		X			X		X

Figure A-3 (Page 3 of 4). Processing Control DDS Keywords

DDS Keyword	APPC	SNUF	BSCCL	Asyn-chronous	Intra-system	Finance	Retail
RCVCONFIRM ² Indicates that the remote program is requesting a confirmation of transaction activity.	X	X			X		
RCVDETACH ² Indicates that the remote program has ended the transaction.	X	X	X		X		X
RCVENDGRP ² Indicates the end of a user-defined group of records sent to the program.		X	X		X	X	X
RCVFAIL ² Indicates that the remote program issued a fail.	X			X	X		
RCVFMH ² Indicates to the program that FMH data has been received.		X			X	X	X
RCVNEGRSP ² Indicates that the remote program issued a negative-response request.		X			X	X	X
RCVTRNRND ² Indicates that the program is now in send state.	X	X	X		X		
RECID ² Used to allow the data content to identify the record format to use to receive the data.	X	X	X	X	X	X	X
RQSWRT Specifies that the program is requesting permission to write.	X	X	X		X		
RSPCONFIRM ² Sends a positive response to a received confirm request.	X	X			X		
SECURITY Includes security information needed to start a program on the remote system.	X	X	X	X	X		X
SUBDEV ² Specifies the subdevice to which output should be directed (for example, printer, punch, and so on).			X		X		

Figure A-3 (Page 4 of 4). Processing Control DDS Keywords

DDS Keyword	APPC	SNUF	BSCCL	Asyn- chronous	Intra- system	Finance	Retail
SYNLVL ² Indicates the synchroniza- tion level of the program.	X				X		
TIMER Allows the user to specify an interval of time to wait before a read-from-invited- program-devices operation receives a timer-expired return code.	X	X	X	X	X	X	X
VARBUFMGT ^{2, 4} Allows the user to send or receive multiple or partial records, rather than just one record, by using one record format per write or read operation.	X						
VARLEN Specifies the length of the data record sent with each write operation.	X	X	X	X	X	X	X
¹ Use of this keyword is restricted. Refer to the <i>Asynchronous Communications Programmer's Guide</i> for more details. ² These DDS keywords do not have system-supplied format equivalents. ³ These keywords are not valid for the 3694 controller. Refer to the <i>Finance Communications Programmer's Guide</i> for details. ⁴ Use of this keyword is restricted. Refer to the <i>APPC Programmer's Guide</i> for more details.							

System-Supplied Format Support

Figure A-4 defines the system-supplied formats supported for ICF, and shows the communications types that supports these formats.

Figure A-4. System-Supplied Format Support

Operation	APPC	SNUF	BSCCL	Asyn- chronous	Intra- system	Finance	Retail
\$\$\$CANL Cancel with invite		X			X	X ¹	X
\$\$\$CANLNI Cancel		X			X	X ¹	X
\$\$\$CNLINV Cancel invite		X	X	X	X	X	X
\$\$\$EOS End of session	X	X	X	X	X	X	X
\$\$\$EVOK Evoke with invite	X	X	X	X	X		X
\$\$\$EVOKET Evoke with detach	X	X	X	X	X		X
\$\$\$EVOKNI Evoke	X	X	X	X	X		X
\$\$\$FAIL Fail	X	X	X	X	X	X	
\$\$\$NRSP Negative response with invite		X			X	X	X
\$\$\$NRSPNI Negative response		X			X	X	X
\$\$\$POSRSP Positive response		X					
\$\$\$RCD Request write with invite	X	X	X		X		
\$\$\$SEND Send with invite or invite	X	X	X	X	X	X	X
\$\$\$SENDE Send with end of group		X	X		X	X	X
\$\$\$SENDET Send with detach	X	X	X		X		X
\$\$\$SENDFM Send FMH with invite		X			X	X	X
\$\$\$SENDNF Send FMH		X		X	X	X	X
\$\$\$SENDNI Send	X	X	X	X	X	X	X
\$\$\$TIMER Timer	X	X	X	X	X	X	X

¹ These keywords are not valid for the 3694 controller. Refer to the *Finance Communications Programmer's Guide* for more information.

Appendix B. Communications Error Handling

This appendix describes programming considerations for ICF communications error recovery. It includes information on:

- System error classifications
- System messages sent on communications errors, and related file error handling in the affected job
- Major/minor return codes and descriptions
- Error reason codes for failed program start requests

System Error Classification

The system divides communications error conditions into several classifications and processes them according to those classifications. The system automatically tries recovery for many of these errors without notification to the using program. In some cases, messages indicating error recovery is in progress are issued to the system operator message queue (QSYSOPR), to the job log, and to other queues specified during device configuration. When the system retry limits specified in configuration objects are exceeded, a message is sent to these queues and any jobs currently using the failing line, controller, or device.

Some errors, such as an application program violation of a communications protocol, do not cause messages to be sent to system message queues, but are reported to the affected program.

It is recommended that all communications programs examine return codes after each operation to detect error conditions and other normal conditions, such as receipt of the detach indication from the remote system. Although many error conditions are reported to the affected job through messages, the primary method for a program to detect these conditions is through return codes and the open feedback and input/output (I/O) feedback areas.

For a complete description of error classifications and system-provided recovery support, see the *Communications Management Guide*.

System Messages

Some errors can occur that do not affect your program. For example, the varying on of a communications line may fail before starting any programs that use devices on the failed line.

Errors that affect your program can occur:

- When a file is opened
- During I/O operations to the file
- When a program device is acquired or released
- When the file is closed

When you encounter errors that can affect the running of a program, a system message is sent to the program message queue of the program using the file.

Error messages are divided into the following message types:

- Notify
- Status
- Diagnostic
- Escape

See the *CL Programmer's Guide* for more information about the message types.

Figure B-1 is a summary of the messages, by operation, that can be issued.

Figure B-1. File Error Message Identifier Groups

Operation	Message Type	Message Identifiers
Open	Diagnostic and status	CPF4001 through CPF4099
Open	Escapes that make the file unusable	CPF4101 through CPF4399
Close	Diagnostic and status	CPF4401 through CPF4499
Close	Escapes that make the file unusable	CPF4501 through CPF4699
Input/Output, Acquire, and Release	Notify with a default reply of cancel, status, and escapes that do not make the file or program device unusable	CPF4701 through CPF4899 and CPF5001 through CPF5099
Input/Output, Acquire, and Release	Notify with a default reply of ignore	CPF4901 through CPF4999
Input/Output, Acquire, and Release	Escapes that make the file or program device unusable	CPF5101 through CPF5399 and CPF5501 through CPF5699

These messages are logged in the job log, and the error is communicated to the program through language status codes and an ICF major/minor return code in the I/O feedback area of the file.

Some conditions are considered normal application exceptions and do not cause job messages. As a result, file error handling for high-level languages is not called. You may need to examine the I/O feedback area for major/minor return codes or other device-specific information. You can detect some conditions by using data description specifications (DDS) response keywords.

User Program Error Detection

All user programs should detect error conditions and determine appropriate error processing. Review all major/minor return codes described in this chapter, and in the programming manual for the communications type you are using, to determine what processing to do.

Permanent errors can cause the session, your program, or both to end. A program can try to recover from errors without ending. The operation you use and the major code you receive determine how your program recovers from the errors.

In general, you recover from an open operation that fails as follows:

- Close the file
- Correct the problem
- Issue the open operation again

An acquire failure is handled as follows:

- Correct the problem
- Issue the acquire operation again

You can resume communications for most I/O operations that encounter session errors and complete with a major return code of 81 by reacquiring the program device associated with the session. For input/output operations that encounter system or file errors completed with a major return code of 80, you may or may not need to close and reopen the file to resume communications. In some cases, depending on the cause of the error, the device must be varied off, then on again, to remedy the problem. Reacquiring the session that failed may also allow you to resume communications. To determine specific error recovery procedures, check each major/minor return code description in the programming manual for the communications type you are using.

If an I/O operation completes with an exception or a nonpermanent error (04, 08, 11, 34, and 83 majors), then the session is still intact and the program can recover, based on the action described for the major/minor return code.

A release failure can be handled in one of two ways. If you want to end the session gracefully, correct the problem as indicated by the return code, and issue the release operation again. For example, if the release operation completes with an 832F, issue a detach request, and then issue the release again. If you want to force the session to end, issue an end-of-session function.

If a close operation fails, issue the close again. A second close is always successful.

The ICF file is implicitly closed when the job ends, if the job ends without recovering from a failure.

Control Language (CL) Commands for Determining Configuration Status

The Work with Configuration Status (WRKCFGSTS) command provides line, controller, device, and mode status for communications on your AS/400 system. Information provided by this command can help you determine the status of your devices and sessions in order to determine error recovery options. The RTVCFGSTS command is also available to determine line, controller, and device status. See the *Communications Management Guide* for more information on configuration status.

Major/Minor Return Codes

This section contains:

- Return code tables that identify all communications types and the return codes that are valid for each. This summary table is useful when you want to make changes to a program so you can use it with a different communications type.

- Summary descriptions of all major and minor return codes for all communications types.

These return codes are set in the I/O feedback area of the ICF file, and report the results of each I/O operation issued by your application program. Your program should check the return code and act accordingly. Refer to your high-level language manual for more information on how to access these return codes.

Each return code is a 4-digit hexadecimal value. The first two digits contain the *major code*, and the last two digits contain the *minor code*.

Notes:

1. In the return code descriptions, *your program* refers to the AS/400 application program that issues the operation and receives a return code from ICF. The *target program* refers to the application program on the remote system with which your program is communicating through ICF.
2. Each communications programming manual provides detailed information about every return code for the communications type and the recovery actions that should be taken.
3. Certain return codes describe the turnaround indication, which is not applicable to asynchronous, retail and finance communications. Also, not all return codes have the same meaning for all communications types.

Major Code 00

A major return code 00 indicates that the operation completed successfully.

Description

The operation issued by your program completed successfully. Your program may have sent or received some data, or may have received a message from the remote system.

Figure B-2 (Page 1 of 2). Major Code 00

Code	APPC	Asyn-chronous	BSECL	Finance	Intra-system	Retail	SNUF
0000	X	X	X	X	X	X	X
0001	X		X	X	X	X	X
0003				X	X	X	X
0004		X			X		X
0005					X	X	X
0007				X	X	X	X
0008	X		X		X		X
000C							X
0010	X		X		X		X
0014	X				X		
0015	X				X		
0016		X					
0017					X		

Figure B-2 (Page 2 of 2). Major Code 00

Code	APPC	Asyn-chronous	BSECL	Finance	Intra-system	Retail	SNUF
001C	X				X		
0020			X				X
0021			X				X
0023							X
0025							X
0027							X
0028			X				X
0030			X				X
0031			X				X
0033							X
0035							X
0037							X
0038			X				X
0042		X					
0044					X		
0045					X		
0047					X		

Code	Description
0000	Turnaround or end-of-transmission indication and data received on successful input operation, or an output operation was successful.
0001	Successful input operation. A turnaround or end-of-transmission indication was not received. Continue to receive.
0003	End-of-group indication received on successful input operation.
0004	Function-management-header and turnaround indications received on successful input operation, or a PAD message received from a remote PAD.
0005	Function-management-header indication received on successful input operation.
0007	Function-management-header and end-of-group indications received on successful input operation.
0008	Detach indication received on successful input operation.
000C	Function-management-header and detach indications received on successful input operation.
0010	Request-to-write, reverse-interrupt (RVI), or request-to-change-direction received on successful output operation.
0014	Turnaround indication received on successful input operation. In addition, the remote system requested confirmation.
0015	Remote system requested confirmation on successful input operation. The local application program continues to receive data.

- 0016** Parity error or stop bit error (framing) or both received on successful input operation.
- 0017** End-of-group indication received on successful input operation. In addition, the remote system requested confirmation.
- 001C** Detach indication received on successful input operation. In addition, the remote system requested confirmation.
- 0020** System message and turnaround or end-of-transmission indication received on a successful input operation.
- 0021** System message received on successful input operation. Continue to receive.
- 0023** System message with end-of-group indication received.
- 0025** Function-management-header indication received with system message.
- 0027** System message received with function-management-header and end-of-group indications.
- 0028** System message and detach indication received on successful input operation.
- 0030** Truncated system message and turnaround or end-of-transmission indication received on successful input operation.
- 0031** Truncated system message received on successful input operation. Continue to receive.
- 0033** Truncated system message received with end-of-group indication.
- 0035** Truncated system message received with function-management-header indication.
- 0037** Truncated system message received with function-management-header and end-of-group indications.
- 0038** Truncated system message and detach indication received on successful input operation.
- 0042** Some data was lost on successful input operation.
- 0044** Function-management-header and turnaround indications received on successful input operation. In addition, the remote system requested confirmation.
- 0045** Function-management-header indication received on successful input operation. In addition, the remote system requested confirmation.
- 0047** Function-management-header and end-of-group indications received on successful input operation. In addition, the remote system requested confirmation.

Major Code 02

The major return code 02 indicates that the input operation completed successfully, but your job is being ended (controlled).

Description

The input operation issued by your program completed successfully. Your program may have received some data or a message from the remote system. However, your job is being ended (controlled).

Figure B-3. Major Code 02

Code	APPC	Asyn-chronous	BSCSEL	Finance	Intra-system	Retail	SNUF
0200	X	X		X	X	X	X
0201	X		X		X		X
0203				X	X	X	X
0204		X			X		X
0205					X	X	X
0207				X	X	X	X
0208	X		X		X		X
020C							X
0214	X				X		
0215	X				X		
0216		X					
0217					X		
021C	X				X		
0220			X				X
0221			X				X
0223							X
0225							X
0227							X
0228			X				X
0230			X				X
0231			X				X
0233							X
0235							X
0237							X
0238			X				X
0242		X					
0244					X		
0245					X		
0247					X		

Code Description

0200 On a successful input operation, a turnaround indication or data that is the beginning or middle record of a group of records was received. In addition, a job ended (controlled) indication was received.

0201 Data with job ended (controlled) indication received on a successful input operation. A turnaround indication was not received. Continue to receive.

- 0203** End-of-group indication received with job ended (controlled) indication on successful input operation.
- 0204** Function-management-header and turnaround indications or a PAD message from a remote PAD was received with job ended (controlled) indication on a successful input operation.
- 0205** Function-management-header indication received with job ended (controlled) indication on successful input operation.
- 0207** Function-management-header and end-of-group indications received with job ended (controlled) indication on successful input operation.
- 0208** Detach indication received with job ended (controlled) indication on successful input operation.
- 020C** Function-management-header and detach indications received with job ended (controlled) indication on successful input operation.
- 0214** Turnaround indication received with job ended (controlled) indication on successful input operation. In addition, the remote system requested confirmation.
- 0215** Remote system requested confirmation. The local application program continues to receive data. Job ended (controlled) indication received.
- 0216** Parity error or stop bit error (framing) or both received with job ended (controlled) indication on successful input operation.
- 0217** End-of-group indication received with job ended (controlled) indication on successful input operation. In addition, the remote system requested confirmation.
- 021C** Detach indication received with job ended (controlled) indication on successful input operation. In addition, the remote system requested confirmation.
- 0220** Remote system message and turnaround or end-of-transmission indication received with job ended (controlled) indication on successful input operation.
- 0221** Remote system message received with job ended (controlled) indication on successful input operation. The session is still in receive state.
- 0223** Remote system message with end-of-group and job ended (controlled) indications received on successful input operation.
- 0225** Function-management-header indication received with system message and job ended (controlled) indication on a successful input operation.
- 0227** System message received with function-management-header, end-of-group, and job ended (controlled) indications on a successful input operation.
- 0228** System message and detach indication received with job ended (controlled) indication on a successful input operation.
- 0230** Truncated system message and turnaround or end-of-transmission indication received with job ended (controlled) indication on successful input operation.
- 0231** Truncated system message received with job ended (controlled) indication on a successful input operation. The session is still in receive state.

- 0233** Truncated system message received with end-of-group and job ended (controlled) indications on a successful input operation.
- 0235** Truncated system message received with function-management-header and job ended (controlled) indications on a successful input operation.
- 0237** Truncated system message received with function-management-header, end-of-group, and job ended (controlled) indications on a successful input operation.
- 0238** Truncated system message and detach indication received with job ended (controlled) indication on successful input operation.
- 0242** Data-loss indication received with job ended (controlled) indication on successful input operation.
- 0244** Function-management-header and turnaround indications with job ended (controlled) indication received on a successful input operation. In addition, the remote system requested confirmation.
- 0245** Function-management-header with job ended (controlled) indication received on a successful input operation. In addition, the remote system requested confirmation.
- 0247** Function-management-header and end-of-group indications with job ended (controlled) indication received on a successful input operation. In addition, the remote system requested confirmation.

Major Code 03

Major return code 03 indicates that the input operation completed successfully, but no data was received.

Description

The input operation issued by your program completed successfully, but no data was received.

Figure B-4. Major Code 03

Code	APPC	Asyn-chronous	BSCCL	Finance	Intra-system	Retail	SNUF
0300	X	X	X		X	X	X
0301	X		X		X		X
0302		X			X		
0303				X	X	X	X
0308	X		X		X	X	X
0309	X	X	X	X	X	X	X
0310	X	X	X	X	X	X	X
0314	X				X		
0315	X				X		
0317					X		
031C	X				X		

Code	Description
0300	On a successful input operation, a turnaround or end-of-transmission indication with no data, or a null record that was the beginning or middle record in a group of records was received.
0301	No data received on successful input operation. A turnaround indication was not received. Continue to receive.
0302	Fail indication received with no data on successful input operation.
0303	End-of-group indication received with no data on successful input operation.
0308	Detach indication received with no data on successful input operation.
0309	Job ended (controlled) indication received on read-from-invited-program-devices operation.
0310	Timer interval has ended.
0312	No data was received, but data from other sources is waiting to be received.
0314	Turnaround indication received with no data on a successful input operation. In addition, the remote system requested confirmation.
0315	Remote system requested confirmation. The local application program continues to receive data.
0317	End-of-group indication received with no data on successful input operation. In addition, the remote system requested confirmation.
031C	A detach indication received with no data on a successful input operation. In addition, the remote system requested confirmation.

Major Code 04

Major return code 04 indicates that an output exception occurred.

Description

An output exception occurred because your program attempted to send data when it should be receiving data. The data from your output operation was not sent. You can attempt to send data later.

Figure B-5. Major Code 04

Code	APPC	Asyn-chronous	BSCCL	Finance	Intra-system	Retail	SNUF
0402	X				X		
0411			X				
0412	X	X	X	X	X	X	X

Code	Description
0402	Fail indication received. Your program must receive.
0411	Message for your program is waiting to be received.
0412	Data for your program is waiting to be received, or a negative response has been received from the remote system, and your program has not been informed.

Major Codes 08 – 11

Major return codes 08 – 11 indicate that miscellaneous program errors occurred.

Description

The operation just attempted by your program was not successful. The operation may have failed because it was issued at the wrong time.

Figure B-6. Major Codes 08-11

Code	APPC	Asyn-chronous	BSCCL	Finance	Intra-system	Retail	SNUF
0800	X	X	X	X	X	X	X
1100	X	X	X	X	X	X	X

Code Description

- 0800** Acquire operation was not successful because your program tried to acquire a program device that has already been acquired.
- 1100** Read-from-invited-program-devices operation was not successful.

Major Code 34

Major return code 34 indicates that an input exception occurred.

Description

The input operation attempted by your program was not successful. The data received was too long for your program's input buffer or was not compatible with the record format specified on the input operation.

Figure B-7. Major Code 34

Code	APPC	Asyn-chronous	BSCCL	Finance	Intra-system	Retail	SNUF
3401			X	X	X	X	X
3431	X						
3441	X	X	X	X	X	X	X
3451	X	X	X	X	X	X	X
3461	X						

Code Description

- 3401** Input operation rejected because data received was too long for your program's input buffer.
- 3431** An input exception occurred because the program received data that exceeded its maximum record length. The data has been truncated. The local application program continues to receive data.
- 3441** The record format selected by the format selection option does not match the record format specified on the read.
- 3451** The file record size specified is not large enough for the data and/or indicators received.

3461 Partial record received because remote system sent an error condition before completing the record.

Major Code 80

Major return code 80 indicates a permanent system or file error (nonrecoverable).

Description

A nonrecoverable file or system error has occurred. The underlying communications support may have ended and your session has ended. If the underlying communications support ended, it must be established again before communications can resume. Recovery from this error is unlikely until the problem causing the error is detected and corrected.

Figure B-8. Major Code 80

Code	APPC	Asynchronous	BSCCEL	Finance	Intra-system	Retail	SNUF
8081	X	X	X	X	X	X	X
8082	X	X	X	X	X	X	X
80B3	X	X	X	X	X	X	X
80C0	X						
80D0	X						
80EB	X	X	X	X	X	X	X
80ED	X	X	X	X	X	X	X
80EF	X	X	X	X	X	X	X
80F8	X	X	X	X	X	X	X

Code Description

- 8081** System error abnormally ended the support provided by the communications type.
- 8082** Communications device not usable or error recovery canceled by operator.
- 80B3** ICF file not available.
- 80C0** Session failed.
- 80D0** Remote transaction program not available. No retry allowed.
- 80EB** Open operation was tried but was not successful. Either an open option was specified that was not valid, or there is a mismatch between the file and the program.
- 80ED** File level check error occurred on open operation.
- 80EF** User not authorized to file.
- 80F8** Open operation not successful because the file is already open or it is in error.

Major Code 81

Major return code 81 indicates a permanent session error (nonrecoverable).

Description

A nonrecoverable session error occurred during an I/O operation. Your session cannot continue and has ended. Before communications can resume, the session must be established by using an acquire operation or another program start request. Recovery from this error is unlikely until the problem causing the error is detected and corrected. Operations directed to other sessions associated with the file should be expected to work.

Figure B-9. Major Code 81

Code	APPC	Asyn-chronous	BSCCL	Finance	Intra-system	Retail	SNUF
810A			X				
8140	X	X	X	X	X	X	X
8187			X				
8191	X	X	X	X		X	X
8192			X				
8193			X				
8194			X				
8196	X						X
8197	X		X	X		X	
8198			X				
8199			X				
819A			X				
819C			X				
819D			X				X
81A3				X		X	
81A4				X		X	
81AD				X		X	
81B9							X
81BA				X		X	
81C2	X						
81C5	X						
81C6	X						
81E9	X	X	X	X	X	X	X

Code Description

- 810A** Combination of values detected on an input or output operation was not valid. Both CODE(ASCII) and TRNSPY(*YES) were specified.
- 8140** Cancel reply was received for a previous inquiry or notify message.
- 8187** Block length or record length is greater than buffer size on an input or output operation.

- 8191** Permanent line error occurred on an output operation, or station (controller) error occurred on an input or output operation.
- 8192** Permanent line error occurred on an input operation.
- 8193** Disconnect indication (for switched lines only) received on an output operation.
- 8194** Disconnect indication (for switched lines only) received on an input operation.
- 8196** Communications support has ended the session.
- 8197** Remote system abnormally ended the session on an output operation.
- 8198** Remote system abnormally ended the session on an input operation.
- 8199** Time between successive data blocks sent to, or received by, the remote system on output operations is larger than specified wait time.
- 819A** Time between successive data blocks received from the remote system on input operations is larger than specified wait time.
- 819C** On an input operation, the length of the data block sent by the remote system was greater than the buffer size.
- 819D** Unexpected data or an unexpected program start request was received from the remote system during an active session.
- 81A3** SNA session ended abnormally.
- 81A4** SNA protocol violation occurred.
- 81AD** Attempt to establish an SNA session was not successful. The SDLC frame size was not large enough to contain the response unit (RU) size.
- 81B9** A data record that exceeds the maximum user record length was received on an input operation.
- 81BA** A data record that exceeds the maximum user record length was received on an input operation.
- 81C2** Operation failed because the local APPC could not establish a session.
- 81C5** The remote program or remote system abnormally ended the session (TYPE = SVC).
- 81C6** The remote program or remote system abnormally ended the session (TYPE = TIMER).
- 81E9** Data received does not match any record format in the file with the RECID keyword.

Major Code 82

Major return code 82 indicates that the open or acquire operation failed.

Description

Your attempt to establish a session was not successful. The error may be recoverable or permanent, and recovery from it is unlikely until the problem causing the error is detected and corrected.

Figure B-10 (Page 1 of 2). Major Code 82

Code	APPC	Asyn-chronous	BSCSEL	Finance	Intra-system	Retail	SNUF
8209	X	X	X	X	X	X	X
820A			X				
8221				X		X	
8233	X	X	X	X	X	X	X
8281	X	X	X	X	X	X	X
8282	X	X	X	X	X	X	X
8285		X					X
8287			X				
8289			X				
828B			X				
828C			X				
828D			X				
828E			X				
8290			X				
8291			X	X		X	X
8293			X				
8297			X	X		X	
82A0			X				
82A1							X
82A2				X			
82A4				X		X	
82A5							X
82A6	X			X		X	X
82A7			X	X		X	X
82A8	X	X	X	X	X	X	X
82A9	X	X	X	X	X	X	X
82AA	X	X	X	X	X	X	X
82AB	X	X	X	X	X	X	X
82AC			X				
82AD				X		X	
82B3	X	X	X	X		X	X
82B4							X
82B5							X
82BB							X
82C3	X						
82EA	X	X	X	X	X	X	X
82EC	X			X		X	
82EE	X	X	X	X	X	X	X
82EF	X	X	X	X	X	X	X

Figure B-10 (Page 2 of 2). Major Code 82							
Code	APPC	Asyn- chronous	BSCSEL	Finance	Intra- system	Retail	SNUF
82F0	X	X					
82F2	X						
82F4	X			X	X	X	
82F5		X	X				X

Code Description

- 8209** An open or acquire operation was not successful because a prestart job is to be ended.
- 820A** Combination of values detected was not valid. Both CODE(ASCII) and TRNSPY(*YES) were specified, or BLOCK(*USER) and RMTBSCSEL(*YES) were specified.
- 8221** SNA command received for remote location or device description that was not supported or not valid.
- 8233** Program device name is either missing or not valid.
- 8281** System error abnormally ended the support provided by the communications type.
- 8282** Communications device not usable or error recovery canceled.
- 8285** Attempt to automatically call remote system failed.
- 8287** Block Length or record length greater than buffer size.
- 8289** Combination of values detected during an acquire operation was not valid. A record separator and text transparency were both specified.
- 828B** Combination of values detected during an acquire operation was not valid. The maximum user record length specified was greater than the block length.
- 828C** Combination of values detected during an acquire or open operation was not valid. GRPSEP(*DEV3740) and BLOCK(*ITB) were both specified.
- 828D** Combination of values detected during an acquire or open operation was not valid. TRUNC(*YES) and BLOCK(*ITB) were both specified.
- 828E** Combination of values detected during an acquire or open operation was not valid. TRUNC(*YES) and BLOCK(*ITB) were both specified, or TRUNC(*YES) and BLOCK(*NOSEP) were specified.
- 8290** Combination of values detected during an acquire or open operation was not valid. Blank compression and text transparency were both specified.
- 8291** Permanent line error or station (controller) error occurred on an unsuccessful open or acquire operation.
- 8293** Disconnect indication (for switched lines only) received from remote system during an acquire or open operation.
- 8297** Remote system ending line transmission.
- 82A0** A record separator character that was not valid was specified on the ADDICFDEVE or OVRICFDEVE command.

- 82A1** Logon portion of the acquire operation failed. Either the host subsystem was not active, or a remote program name that was not valid was specified in the APPID parameter.
- 82A2** User ID or password that was not valid was received on the INIT-SELF.
- 82A4** SNA protocol violation occurred.
- 82A5** Combination of parameter values detected during an acquire operation was not valid. *YES was specified for the MSGPTC and BATCH parameters.
- 82A6** SNA bind command failed.
- 82A7** The specified program device was already in use when the open or acquire operation was attempted.
- 82A8** The maximum number of program devices allowed for the ICF file was reached when the open or acquire operation was attempted.
- 82A9** Acquire to requesting program device rejected because *REQUESTER device was not available, was already acquired, or a CPI-Communications requesting conversation was already allocated.
- 82AA** Operation failed because remote location or device not found, or device was found but not usable.
- 82AB** Operation failed because device not varied on.
- 82AC** Acquire operation failed because the remote location specified for the device was not *REQUESTER.
- 82AD** Attempt to establish an SNA session was not successful. The SDLC frame size was not large enough to contain the RU size.
- 82B3** Operation failed because device is being used by a different job or no sessions are currently available for specified remote location.
- 82B4** Operation failed because System/36 application program cannot open an ICF file to a program device for SNA 3270 program interface.
- 82B5** Operation not successful because SNA 3270 program interface session cannot use SNUF device from earlier release.
- 82BB** Acquire operation failed because device specified was reserved for a program start request from host system.
- 82C3** Mode description was not found.
- 82EA** *RECID format selection processing was requested to a file that contains no record formats with a *RECID keyword.
- 82EC** The acquire operation was not successful because CNVTYPE(*USER) is not valid with FMTSLT(*RMTFMT); or this communications type does not support FMTSLT(*RMTFMT).
- 82EE** An operation was attempted to a device that is not supported for an ICF file.
- 82EF** An open or acquire operation was attempted to a device the user is not authorized to use, or to a device in service mode.
- 82F0** Error recovery not performed for file.
- 82F2** Conversation type specified for the requesting program device does not match value received from source program.

82F4 Open operation for input only not valid for a source program.

82F5 *RMTFMT format selection parameter not valid on acquire operation.

Major Code 83

Major return code 83 indicates that a session error occurred (the error is recoverable).

Description

An error occurred during an I/O operation, but the session is still active. Recovery within your program might be possible.

Figure B-11 (Page 1 of 2). Major Code 83

Code	APPC	Asyn-chronous	BSCCL	Finance	Intra-system	Retail	SNUF
830B	X	X	X	X	X	X	X
830C							X
830D							X
8311							X
8316	X						
8319				X	X	X	X
831A			X		X	X	
831B				X	X	X	X
831C		X	X	X	X	X	X
831E	X	X	X	X	X	X	X
831F	X	X	X	X	X	X	X
8322			X	X	X	X	X
8323				X	X		X
8324							X
8326				X	X	X	X
8327	X		X		X	X	X
8329	X	X	X		X	X	X
832A					X		
832B			X				
832C	X	X	X	X	X	X	X
832D	X	X	X	X	X	X	X
832F	X		X	X	X	X	X
8330					X	X	X
8331					X		X
8332							X
8334	X		X		X	X	
83B6				X		X	X
83B7							X
83B8							X

Figure B-11 (Page 2 of 2). Major Code 83

Code	APPC	Asyn-chronous	BSECL	Finance	Intra-system	Retail	SNUF
83C7	X						
83C8	X						
83C9	X						
83CA	X						
83CB	X						
83CC	X						
83CD	X				X		
83CE	X						
83CF	X						
83D0	X						
83D1	X						
83D2	X						
83D3	X						
83D5	X						
83D6	X				X		
83E0	X	X	X	X	X	X	X
83E8		X	X	X	X	X	X
83F1	X						
83F3	X						
83F6			X				
83F7			X				
83F8	X	X	X	X	X	X	X
83F9	X						

Code Description

- 830B** An input or output operation was attempted to a program device that was not acquired.
- 830C** Length of function-management-header received from host system is greater than the maximum RU length.
- 830D** Shutdown indication received from host system.
- 8311** Output operation was attempted while a message containing sense data was waiting to be received.
- 8316** Evoke failed because target program was not found.
- 8319** Negative response with sense data was issued to your program's previous or current output request by the other program.
- 831A** Evoke failed to complete successfully, or the target program ended abnormally.
- 831B** Sense data that was not valid was specified on a negative-response function issued by your program.

- 831C** Operation is not valid at this time. An indication was received that the return code from a previous operation was not properly handled by your program.
- 831E** Operation or combination of operations not valid, or operation not supported by the communications type.
- 831F** Length of data record or data specified on the operation not valid.
- 8322** Request-to-write, negative-response, or detach function not valid while your program is in send state.
- 8323** Cancel issued while in receive state, or fail indication received while in send state.
- 8324** Function-management-header indication issued by your program at wrong time. Function-management-header is valid only with the *first* record in the chain.
- 8326** Cancel or negative-response indication issued as a single record. These functions are only valid within a group of records.
- 8327** Input or output operation that was not valid was issued when no transaction existed. Your program may have expected more data when there was none.
- 8329** Evoke not valid in this session. Your program was started by a remote program start request.
- 832A** Both programs were attempting to receive.
- 832B** Output operation with zero record length detected when GRPSEP(*OFCSYS) was specified.
- 832C** Release operation not valid after an invite function.
- 832D** Attempted function is not valid following an invite function.
- 832F** Evoke function or release operation that was not valid was issued before a transaction completed.
- 8330** Cancel indication or cancel and turnaround indications received on an input operation.
- 8331** Cancel indication received *without* turnaround indication on an input operation.
- 8332** Cancel and detach indications received on an input operation.
- 8334** Program name missing on an evoke sent by your program, or the length of the program name was not valid.
- 83B6** The remote program has quiesced the SNA session on which this transaction is running.
- 83B7** Value received in SNA header that was not valid.
- 83B8** The host system has sent a clear request to reset the session.
- 83C7** Fail indication (TYPE=PROG) received with no data on an input operation. No data truncated.
- 83C8** Fail indication (TYPE=SVC) received with no data on an input operation. No data truncated.
- 83C9** Fail indication (TYPE=PROG) received on an input or output operation with or without a confirm indication. Data may have been lost.

- 83CA** Fail indication (TYPE=SVC) received on an input or output operation with or without a confirm indication. Data may have been lost.
- 83CB** Fail indication (TYPE=PROG) received on an input operation. The last logical record truncated.
- 83CC** Fail indication (TYPE=SVC) received on an input operation. The last logical record truncated.
- 83CD** Confirm indication not allowed when SYNLVL(*NONE) is specified on the evoke function.
- 83CE** Security information specified on the evoke function not valid. Request rejected by remote system.
- 83CF** Remote location or remote program does not support the specified conversation type. Request rejected by remote system.
- 83D0** Program name specified on the evoke function is not currently available. Retry is allowed.
- 83D1** Program initialization parameters not allowed. Request rejected by remote system.
- 83D2** Program initialization parameters not specified correctly. Request rejected by remote system.
- 83D3** Synchronization level specified on the evoke function not supported by remote program.
- 83D5** Response-to-confirm request required.
- 83D6** Response-to-confirm request not valid in current state.
- 83E0** Record format not defined for the file.
- 83E8** Cancel-invite function not valid because an invite function was not previously issued.
- 83F1** The file was closed while the transaction was still active.
- 83F3** Length specification on a basic conversation not valid.
- 83F6** User-defined data not valid on an unsuccessful output operation.
- 83F7** Length of user-blocked data record not valid on an unsuccessful output operation.
- 83F8** Operation attempted to a device marked in error.
- 83F9** Your program issued an operation that did not complete the data record.

Failed Program Start Requests

Message CPF1269 is sent to the system operator message queue when the local system rejects an incoming program start request. You can use the message information to determine why the program start request was rejected.

The CPF1269 message contains two reason codes. One of the reason codes can be zero, which can be ignored. If only one nonzero reason code is received, that reason code represents the reason the program start request was rejected. If the System/36 environment is installed on your AS/400 system, there can be two nonzero reason codes. These two reason codes occur when the OS/400 system cannot determine whether the program start request was to start a job in System/36 environment or in the OS/400 environment. One reason code

explains why the program start request was rejected in the System/36 environment and the other explains why the program start request was rejected in the OS/400 environment. Whenever you receive two reason codes, you should determine which environment the job was to run in and correct the problem for that environment.

Figure B-12 describes reason codes for failed program start requests.

Figure B-12 (Page 1 of 4). Reason Codes for Rejected Program Start Requests

Reason Code	Reason Description	Communications Types
401	Program start request received to a device that is not allocated to an active subsystem.	All
402	Requested device is currently being held by a Hold Communications Device (HLDCMNDEV) command.	All except APPC
403	User profile is not accessible.	All
404	Job description is not accessible.	All
405	Output queue is not accessible.	All
406	Maximum number of jobs defined by subsystem description are already active.	All
407	Maximum number of jobs defined by communications entry are already active.	All
408	Maximum number of jobs defined by routing entry are already active.	All
409	Library on library list is exclusively in use by another job.	All
410	Group profile cannot be accessed.	All
411	Insufficient storage in machine pool to start job.	All
412	System values not accessible.	All
501	Job description was not found.	All
502	Output queue was not found.	All
503	Class was not found.	All
504	Library on initial library list was not found.	All
505	Job description or job description library is damaged.	All
506	Library on library list is destroyed.	All
507	Duplicate libraries were found on library list.	All
508	Storage-pool defined size is zero.	All
602	Transaction program-name value is reserved but not supported.	All
604	Matching routing entry was not found.	All
605	Program was not found in the library specified.	All
704	Password is not valid.	All except Retail
705	User is not authorized to device.	All
706	User is not authorized to subsystem description.	All
707	User is not authorized to job description.	All
708	User is not authorized to output queue.	All
709	User is not authorized to program.	All

Figure B-12 (Page 2 of 4). Reason Codes for Rejected Program Start Requests

Reason Code	Reason Description	Communications Types
710	User is not authorized to class.	All
711	User is not authorized to library on library list.	All
712	User is not authorized to group profile.	All
713	User ID is not valid.	All except Retail
714	Default user profile is not valid.	All except Finance
715	Neither password nor user ID was provided, and no default user profile was specified in the communications entry.	All
718	No user ID.	All except Retail and APPC
722	A user ID was provided, but no password was sent.	All except Retail
723	No password was associated with the user ID.	All
725	User ID does not follow naming convention.	All except Retail
726	User profile has been disabled.	All
801	Program initialization parameters are present but not allowed.	All except Retail
802	Program initialization parameter exceeds 2000 bytes for a prestart job.	All except Retail
803	Subsystem is ending.	All
804	Prestart job is inactive or is ending.	All
805	WAIT(*NO) was specified on the prestart job entry and no prestart job was available.	All
806	The maximum number of prestart jobs that can be active on a prestart job entry was exceeded.	All
807	Prestart job ended when a program start request was being received.	All
901	Program initialization parameters are not valid.	All except Retail
902	Invalid number of parameters for program.	All
903	Program initialization parameters required but not present.	All
1001	System logic error. Function check or unexpected return code encountered.	All
1002	System logic error. Function check or unexpected return code encountered while receiving program initialization parameters.	All
1501	Character in procedure name is not valid.	All
1502	Procedure not found.	All
1503	S/36 environment library not found.	All
1504	Library QSSP not found.	All
1505	File QS36PRC not found in library QSSP.	All
1506	S/36 Procedure or library name is greater than 8 characters.	All
1507	Current library not found.	All
1508	Not authorized to current library.	All

Figure B-12 (Page 3 of 4). Reason Codes for Rejected Program Start Requests

Reason Code	Reason Description	Communications Types
1509	Not authorized to QS36PRC in current library.	All
1510	Not authorized to procedure in current library.	All
1511	Not authorized to S/36 environment library.	All
1512	Not authorized to file QS36PRC in S/36 environment library.	All
1513	Not authorized to procedure in S/36 environment library.	All
1514	Not authorized in library QSSP.	All
1515	Not authorized to file QS36PRC in QSSP.	All
1516	Not authorized to procedure in QS36PRC in QSSP.	All
1517	Unexpected return code from S/36 environment support.	All
1518	Problem phase program not found in QSSP.	All
1519	Not authorized to problem phase program in QSSP.	All
1520	Maximum number of target programs started (100 per S/36 environment).	All
1901	The record or block size exceeds maximum size.	BSCCEL
1902	ASCII and transparency are mutually exclusive.	BSCCEL
1903	Transparent mode and blank compression conflict.	BSCCEL
1904	Block length is required with data format.	BSCCEL
1905	Blank truncation and ITB mode conflict.	BSCCEL
1906	Blank compression and ITB mode conflict.	BSCCEL
1907	3740 multiple files and ITB mode conflict.	BSCCEL
1908	Record separator mode and transparent modes conflict.	BSCCEL
1909	Record separator and ITB mode conflict.	BSCCEL
1910	The record length exceeds the block length.	BSCCEL
1911	Record separator character is not valid.	BSCCEL
1912	BLOCK(*USER) and RMTBSCCEL(*YES) conflict.	BSCCEL
1913	BLOCK(*NOSEP) and TRUNC(*YES) conflict.	BSCCEL
1914	Program name is not valid.	BSCCEL
1915	Program start request record was too long.	BSCCEL
2001	FMH5 field length is not correct.	APPC
2002	Concatenation code is not valid.	APPC
2003	Function-management-header type is not 5.	APPC
2004	Command code field in function-management-header is not valid.	APPC
2005	Length for fixed-length fields is not valid.	APPC
2006	Conversation type is not supported.	APPC
2007	Synchronization level is not supported.	APPC
2008	Reconnection is not supported.	APPC
2009	Transaction program name field length is not valid.	APPC

Figure B-12 (Page 4 of 4). Reason Codes for Rejected Program Start Requests

Reason Code	Reason Description	Communications Types
2010	Access code subfield length is not valid.	APPC
2011	UOW-ID subfield length is not valid.	APPC
2012	UOW-ID contents are not valid.	APPC
2013	Requested device is currently being held by a Hold Communications Device (HLDCMNDEV) command.	APPC
2014	Transaction program name value is reserved but not supported.	APPC
2015	LU service request received but the LU service job is not active.	APPC
2016	Pre-verified user ID received, but device description specifies SECURELU(*NO).	APPC
2017	No user ID was provided, but a password was received.	APPC
2018	No user ID was provided, but a user profile was received.	APPC
2019	Remote system indicated it sent a pre-verified user ID, but no user ID was received.	APPC
2020	Remote system sent a pre-verified user ID, but also sent a password.	APPC
2021	Remote system sent a user ID (which it had not verified) and failed to send a password.	APPC
2022	Password received, but this is a nonsecure system.	APPC
2111	Program name missing or not valid.	SNUF
2118	Function-management-header not valid.	SNUF
2123	End bracket or end chain missing.	SNUF
2501	System logic error. Function check or unexpected return code encountered while processing a program start request.	Intrasystem
2502	Temporarily unable to allocate needed resources for a program start request.	Intrasystem
2503	No subsystem accepting program start requests for this device.	Intrasystem
2601	Program name missing and device is not configured for display station pass-through.	Retail
2651	*EXEC not specified.	Finance
2652	Blank missing after *EXEC.	Finance
2653	Program name missing.	Finance
2654	Program name greater than 10 characters.	Finance
2655	Library name greater than 10 characters.	Finance

Appendix C. Open Feedback and I/O Feedback

This appendix contains information concerning the open feedback and I/O feedback areas.

and the *Extended Program Model User's Guide and Reference* for more information.

If you are using the COBOL/400, FORTRAN/400, or RPG/400 support, you need to add the offset values shown in Figure C-1 to the offset values listed in this appendix to access information from the feedback areas.

For C/400 programming language, the Open Feedback Area is accessed with a call to the `_Ropnfbk` routine. The Common I/O Feedback Area is accessed with a call to the `_Riofbk` routine. The File Dependent Feedback Area is accessed by adding an offset in the Common I/O Feedback Area to a pointer to the Common I/O Feedback Area. See the *C/400* User's Guide*

Open Feedback Area

You can use the open feedback area information, set during open processing, as long as the file is open. The support provided by the high-level language you are using determines how to access this information. See the appropriate language reference manual for more information.

The complete open feedback area is described in the *Data Management Guide*. Figure C-2 shows the fields relevant to ICF support.

Figure C-1. Offset Values for COBOL/400, FORTRAN/400, and RPG/400

Language	Open Feedback Area	Common I/O Feedback Area	File Dependent Feedback Area
COBOL/400	0 ¹	0	144
FORTRAN/400	0 ¹	0	144 ²
RPG/400	81	241	367

¹ Separate structure from the Common I/O Feedback Area.

² Use offset in the first 2 bytes of the Common I/O Feedback Area to get to the File Dependent Feedback Area.

Figure C-2 (Page 1 of 2). Open Feedback Area

Offset	Data Type	Length in Bytes	Contents
0	Character	2	Type of file being opened (DS = device file).
2	Character	10	Name of the file being opened.
12	Character	10	Name of the library containing the file.
44	Binary	2	Record length.
66	Binary	2	File type (11 = ICF).
116	Character	10	Program device name of the requester. This field contains the program device name, if valid for this job and file. If it is not valid, this field contains *N.
133	Character	2	Open identifier. This value is unique for a full open of a file (SHARE (* NO)) or the first open of a file with (SHARE (* YES)). It allows the user to match this file to an entry on the associated data queue.
135	Binary	2	Maximum record length. This value includes the data, source sequence numbers, option indicators, response indicators, and P-data, if applicable. If this field is 0, then use the field from offset 44.

Figure C-2 (Page 2 of 2). Open Feedback Area

Offset	Data Type	Length in Bytes	Contents
146	Binary	2	Number of program devices added to the file using the Add Inter-system Communications Function Device Entry (ADDICFDEVE) command, or number of program devices defined to the file using the Override ICF Device Entry (OVRICFDEVE) command and acquired by the program.
148	Character	Variable	Program device name definition list. Refer to Figure C-3.

Program Device Definition List

Figure C-3 shows the mapping for a single entry of the program device definition list, which is physically a part of the open feedback area. However, the fields in the definition list are not necessarily set when the file opens.

Figure C-3 (Page 1 of 2). Program Device Definition List

Offset	Data Type	Length in Bytes	Contents
0	Character	10	Program device name.
60	Character	10	Device description name.
70	Character	1	Device class. Hex 0B for ICF.
71	Character	1	Communications type:
	Hex 0A		Binary synchronous communications equivalence link (BSCEL) device
	Hex 0E		Advanced Program-to-Program Communications (APPC) device
	Hex 1E		Intrasystem device
	Hex 1F		Asynchronous device
	Hex 20		System network architecture upline facility (SNUF) device
	Hex 42		Finance
	Hex 43		Retail

Figure C-3 (Page 2 of 2). Program Device Definition List

Offset	Data Type	Length in Bytes	Contents
76	Character	2	Status flags, as follows: Bit 3, acquire status: 0 Program device not acquired. 1 Program device acquired. Bit 4, invite status: 0 Program device not invited. 1 Program device invited. Bit 5, data-available status: 0 Data not available. 1 Data available. Bit 6, transaction status: 0 Transaction not started. An evoke has not been sent, a detach has been sent or received, or the transaction has completed. 1 Transaction started. The session is active, an evoke has been sent or received, and the transaction has not ended. Bit 7, session type: 0 Session created with source program 1 Requesting program device
78	Character	1	Synchronization level: Hex 00 The transaction was started with SYNLVL(*NONE). Confirm processing is not allowed. Hex 01 The transaction was started with SYNLVL(*CONFIRM). Confirm processing is allowed.
79	Character	1	Conversation type: Hex D0 Basic Hex D1 Mapped

Input/Output Feedback Area

The results of I/O operations are communicated to the program using ICF messages and I/O feedback information. The support provided by the high-level language you are using determines how to access this information. See the

appropriate communications language reference manual for more information.

The feedback area consists of two parts:

- A common I/O feedback area
- A file-dependent I/O feedback area

Common I/O Feedback Area

The complete common I/O feedback area is described in the *Data Management Guide*. Figure C-4 shows the fields relevant to ICF.

Figure C-4. Common I/O Feedback Area

Offset	Data Type	Length in Bytes	Contents
0	Binary	2	Offset to device-dependent feedback area. Refer to Figure C-5 on page C-5.
2	Binary	4	Output operation count. Updated only when a output operation completes successfully. This field is not updated for fail, request-to-write, cancel, or negative-response functions, or if the record length is 0.
6	Binary	4	Input operation count. Updated only when an input operation completes successfully and data is received.
10	Binary	4	Output then input operation count. Updated only when a combined output then input operation completes successfully.
14	Binary	4	Count of other operations. Number of successful acquire and release operations.
18	Character	1	Reserved.
19	Character	1	Current operation (the last requested): Hex 01 Input Hex 05 Output Hex 06 Output then Input Hex 11 Release Hex 12 Acquire
20	Character	10	Name of the just-processed record format, which is either specified on the I/O request or determined by default processing.
30	Character	2	Communications class and type: Hex 0B0A BSCCEL Hex 0B0E APPC Hex 0B1E Intrasystem Hex 0B1F Asynchronous Hex 0B20 SNUF Hex 0B42 Finance Hex 0B43 Retail
32	Character	10	Program device name (for operation just completed).
42	Binary	4	Record length specified by the record format processed by the last I/O operation, not including any indicators or program-to-system fields (P-data fields). For Evoke operations, this field specifies the length of any user-defined parameters.
46	Reserved	82	Not applicable to ICF communications.
128	Binary	2	Length of the record associated with the last I/O operation. This value includes data, option indicators, response indicators, and P-data, if applicable.
130	Reserved	14	Not applicable to ICF communications.

File-Dependent I/O Feedback Area

Figure C-5 shows the communications type-dependent fields relevant to ICF.

Figure C-5. File-dependent I/O Feedback Area

Offset	Data Type	Length in Bytes	Contents	ICF Type
5	Binary	4	Actual record length. This field is set as follows: Input: Contains the actual length of user data received from the remote system or device. When all the data cannot be contained in the record format used, the length of data is provided, if known. If the actual length cannot be determined, this field is set to hex FFFFFFFF. When a partial record is received, the length of the data received is provided. If the input operation completes with an error (other than partial record or buffer too small), the contents of the field are undetermined. Output: Contains the number of bytes moved from your program to the output buffer. If the output operation completes with an error, the contents of the field are undetermined.	All
34	Character	2	Major return code.	All
36	Character	2	Minor return code.	All
38	Character	8	Negative-response error data. For some return codes, this field can contain more detailed information about the reason for the error.	APPC Finance Retail
46	Character	1	Safe indicator: 0 Off. 1 On indicates that a block ending with ETX was received. The Safe indicator is not set for BLOCK(*USER).	BSCCL
47	Character	1	Reserved.	
48	Character	1	Request-to-write indication was received. 0 Request-to-write not received. 1 Request-to-write was received.	APPC SNUF BSCCL Intrasystem
49	Character	10	Remote format name received from the remote program on an input operation.	APPC Intrasystem
63	Character	8	Mode associated with the program device.	APPC

Appendix D. EBCDIC and ASCII Character Sets

The following charts show the EBCDIC and ASCII character sets. The charts are provided to show the data link control characters that are used in data communications.

EBCDIC Character Set

Figure D-1 shows a complete EBCDIC character set.

		Main Storage Bit Positions 0,1,2,3															
Main Storage Bit Positions 4,5,6,7	Hex	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE	DS		SP	&	-					{	}	\	0	
0001	1	SOH	DC1	SOS		RSP	/		a	j	~		A	J	NSP	1	
0010	2	STX	DC2	FS	SYN				b	k	s		B	K	S	2	
0011	3	ETX	DC3	WUS	IR				c	l	t		C	L	T	3	
0100	4	SEL	ENP RES	INP BYP	PP				d	m	u		D	M	U	4	
0101	5	HT	NL	LF	RS				e	n	v		E	N	V	5	
0110	6		BS	ETB	NBS				f	o	w		F	O	W	6	
0111	7	DEL	POC	ESC	EOT				g	p	x		G	P	X	7	
1000	8	GE	CAN		SBS				h	q	y		H	Q	Y	8	
1001	9	SPS	EM		IT				'	i	r	z		I	R	Z	9
1010	A	RPT	UBS	SM SW	RFF	¢		:					SHY				
1011	B	VT	CU1	FMT	CU3	.	\$,	#								
1100	C	FF	IFS			<	.	%	@				⌋		⌈		
1101	D	CR	IGS	ENQ	NAK	()	—	'								
1110	E	SO	IRS	ACK		+	;	>	=				⌋				
1111	F	SI	ITB IUS	BEL	SUB		⌋	?	"							EQ	

RSLS106-0

Figure D-1. EBCDIC Character Set

ASCII Character Set

Figure D-2 shows the ASCII character set.

Main Storage Bit Positions 4,5,6,7		Main Storage Bit Positions 0,1,2,3															
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Hex		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE	SP	0	@	P	·	p								
0001	1	SOH	DC1	!	1	A	Q	a	q								
0010	2	STX	DC2	"	2	B	R	b	r								
0011	3	ETX	DC3	#	3	C	S	c	s								
0100	4	EOT	DC4	\$	4	D	T	d	t								
0101	5	ENQ	NAK	%	5	E	U	e	u								
0110	6	ACK	SYN	&	6	F	V	f	v								
0111	7	BEL	ETB	'	7	G	W	g	w								
1000	8	BS	CAN	(8	H	X	h	x								
1001	9	HT	EM)	9	I	Y	i	y								
1010	A	LF	SUB	·	:	J	Z	j	z								
1011	B	VT	ESC	+	;	K	[k	{								
1100	C	FF	FS	,	<	L	\	l									
1101	D	CR	GS	-	=	M]	m	}								
1110	E	SO	RS	.	>	N	~	n	~								
1111	F	SI	US	/	?	O	_	o	DEL								

RSL5197-1

Figure D-2. ASCII Character Set

Appendix E. File Transfer Support

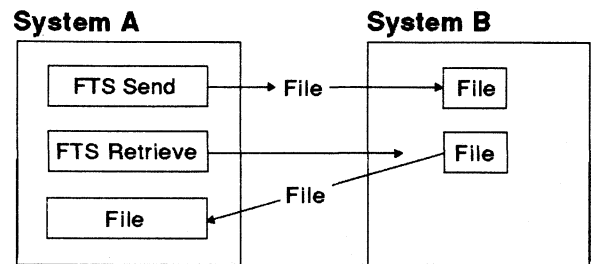
This appendix describes the application interface to the AS/400 system **file transfer support (FTS)**. FTS is a function of the operating system that moves the file members from one system to another using asynchronous, advanced program-to-program communications (APPC), or binary synchronous communications equivalence link (BSC/CL) communications support.

File Transfer Support Overview

Using FTS, a user application program can send or retrieve database file members between one AS/400 system and another AS/400 system, send database file members to System/36, and retrieve files and library members from System/36. System/36 Release 5.1 with preventive PTF – DK3700 is required to communicate with System/36. FTS does not support the sending and retrieving of database file members between an AS/400 system and a System/38.

If an AS/400 database file has more than one member, you can send or retrieve only one member at a time. If a database file does not exist, it is created and the member is added. If a database file exists and the member does not exist, the member is added. If a member exists, you can specify that the member be replaced.

FTS running on the local system communicates with FTS on the remote system to complete the request. FTS defines the **target system** as the system that receives the object, which can be the local system or the remote system. For example, if system A in Figure E-1 sends a database file member to system B using FTS, system B is the target system. If system A retrieves a database file member from system B using FTS support, system A is the target system.



RSL5150-1

Figure E-1. Example of File Transfer Support

An application program can use FTS by calling the program QY2FTML.

Note: FTS uses the file QSYS/QY2ICFF. Do not change or delete this file. This file contains the DDS formats for the ICF file.

Either a high-level language program or a control language (CL) program can call FTS. FTS is supported by all high-level languages. Refer to the C/400, COBOL/400, RPG/400, and CL program examples in this appendix for more details.

FTS evokes a partner application on the remote system. You do not need to have a user application on the remote system to use FTS.

FTS assumes that the user ID used on the remote system is the same as the user ID for the job from which FTS was started. If you are creating a file on an AS/400 system, the sending user ID becomes the owner.

For line speeds greater than or equal to 56000 bits per second (bps), FTS does not compress data. However, for a token ring network, FTS always compresses data. Therefore, for line speeds greater than 56000 bps, FTS can transmit data between an AS/400 system and a System/36 or another AS/400 system that uses a token ring network.

File Transfer Considerations

The following sections describe in more detail the objects that you can send and retrieve using FTS.

To and From an AS/400 System

You can send physical file members to or retrieve them from an AS/400 system.

You cannot send or retrieve the following objects:

- Physical files that are part of an interactive data definition utility (IDDU) dictionary
- Physical files being sent to a receiving AS/400 system, where the receiving AS/400 system has logical files built over the physical files

Note: The logical files must be deleted by the user, then the FTS can successfully transfer the physical files.

- AS/400 program objects
- Logical files
- Device files

AS/400 System Retrieving from System/36

On an AS/400 system, you can retrieve the following objects from System/36:

- System/36 files
- System/36 library members

You cannot retrieve the following objects:

- System/36 data dictionary
- System/36 folder/document

When you retrieve a file from System/36, the file is stored on the AS/400 system as a physical file within a library. If you do not specify a member name, the member name becomes the creation date with an M added as the first character.

When you retrieve a library member from System/36, the file is stored on the AS/400 system as a physical file member in a file within a library. If you do not specify the file name, the type of member determines the file name. Source library members are stored in file QS36SRC, procedure library members are stored in file QS36PRC, subroutine library members are stored in file QS36SBR, and load library members are stored in file QS36LOD.

Note: A System/36 index file cannot be added as a member to a keyed file created on an AS/400 system.

AS/400 System Sending to System/36

You can send physical file members from an AS/400 system to System/36.

You cannot send the following objects:

- Physical files that are part of an interactive data definition utility (IDDU) dictionary
- AS/400 program objects
- Logical files
- Device files

The attributes of the physical file determine if physical file members are stored as library members or files on System/36.

If the physical file member has the attributes of a System/36 file, it is stored as a file on System/36. The file keeps its file organization (such as direct, sequential, or indexed). The file name and date field from the parameters are used to create the file on System/36.

If the physical file member being sent does not have the attributes of a System/36 file, the member is stored on System/36 as a library member. The library name, the member name, and the type field from the input parameters are used to create the library member on System/36.

Notes:

1. If a library name, file name, or member name is greater than 8 characters, System/36 uses the first 8 characters as the name.
2. The AS/400 system receives and keep source members with no records but the System/36 does not. If an empty source member is sent to a System/36, the source member is not kept.

Multiple Communication-Type Support

When using FTS, your system can set up communications with the following communication types and specific links:

- Advanced program-to-program communications (APPC)

- Multiple sessions at the same time
- Switched or nonswitched connections
- Synchronous data link control (SDLC) links
- X.25 links
- Integrated services digital network (ISDN)
- Local area network (LAN) links
 - Token Ring
 - Ethernet
- Advanced peer-to-peer networking (APPN) capability
- Logical unit (LU) own

When you configure a device description, one of the mode names you specify must be *NETATR.

- Binary synchronous communications equivalence link (BSCSEL)
 - Single session only
 - Switched or nonswitched connections

With BSCSEL, you can configure the maximum user record length that you are going to send. This length cannot be greater than 4075. You must specify *YES for the TRNSPY parameter and *YES for the RMTBSCSEL parameter when you configure BSCSEL. BSCSEL supports a maximum length of 8 characters for the user ID and a maximum of 4 characters for the password. You must also specify *NONE for the BLOCK parameter (this is the default value).

- Asynchronous communications
 - Single session only
 - Switched or nonswitched connections

Asynchronous communications performs a logical link protocol to ensure data integrity. Data is sent as 8-bit EBCDIC, not as ASCII. When you are using FTS on an asynchronous line description, your modem must be set to full duplex.

If your system is connected to a network by a packet assembler/disassembler (PAD) and you use FTS on an X.25 packet-switched data network (PSDN),

you must set the X.3 parameters and any network-specific parameters to allow data transparency. To achieve data transparency, set the X.3 parameters and any network-specific parameters to allow the following:

- No PAD recall using a character
- No echo
- No selection of data forwarding characters
- No use of XON/XOFF
- PAD must send interrupt when a break signal from start-stop mode data terminal equipment (DTE) is received
- PAD must allow EBCDIC data
- PAD must allow 8-bit transparency
- Only forward on full packets or idle timer

Notes:

1. The network PAD must not perform any operations on the file transfer data stream.
2. If you use FTS with AS/400 integrated PAD, the X.3 parameter settings are ignored in order to achieve data transparency.

See the *Asynchronous Communications Programmer's Guide* for more information about the X.3 parameters.

You must create configurations for each communication type and vary on the configuration on both the local and remote systems. For further information on communications configurations, see the *OS/400* Communications Configuration Reference* manual. FTS establishes a link to the varied on configuration based on the remote location name supplied in the input parameters.

FTS has a maximum record length of 4075. If you use BSCSEL and configure your maximum user record length, the record must be at least 512 bytes long. You can, however, send files with less than 512-byte records.

File Transfer Parameters

This section describes the parameters passed to the file transfer program QY2FTML. FTS parameters are all positional. You must, therefore, reserve space in your program for all parameters. If you do not use a parameter, fill its space with blanks.

Refer to the *CL Reference* manual for general rules about naming libraries, database files, and

database file members. Refer to the *System/36 System Reference* for System/36 naming conventions.

To and From an AS/400 System

Figure E-2 describes the parameters for sending and retrieving files between one AS/400 system and another AS/400 system.

Figure E-2 (Page 1 of 3). Transferring Files to and from an AS/400 System

Parameter	Value	Description
OPTION	Character	File transfer option to perform. Length: 1 character Type: Input, required Valid values are as follows: <ul style="list-style-type: none">• S—Send• R—Retrieve
FROMLIB	Library name	Name of the library that contains the database file. Length: 10 characters Type: Input, required Valid values: Library name
FROMFILE	File name	Name of the database file that contains the member. Length: 10 characters Type: Input, required Valid values: Database file name
FROMMBR	Member name	Name of the member. Length: 10 characters Type: Input, required Valid values: Member name
TYPE	Blanks	Not needed for an AS/400 system to an AS/400 system. Length: 6 characters Type: Not applicable Valid values: Blanks
TOLIB	Target library name	Name of the receiving library. Length: 10 characters Type: Input, not required Valid values: Library name Default: FROMLIB

Figure E-2 (Page 2 of 3). Transferring Files to and from an AS/400 System

Parameter	Value	Description
TOFILE	Target file name	Name of the receiving database file. Length: 10 characters Type: Input, not required Valid values: File name Default: FROMFILE
TOMBR	Target file member name	Name of the receiving member. Length: 10 characters Type: Input, not required Valid values: Member name Default: FROMMBR When you use TOMBR, consider the following: <ul style="list-style-type: none"> • If you are replacing a database member (REPLACE=Y), this is the name of the member to replace at the target system. • If you are adding a new database member, this is the name to assign.
TODATE	Blanks	Not needed for an AS/400 system to an AS/400 system. Length: 6 characters Type: Not applicable Valid values: Blanks
REPLACE	Character	This field tells whether you want to replace the member on the target system. Length: 1 character Type: Input, not required Valid values: <ul style="list-style-type: none"> • Y—Replace an existing member on the target system. • N—Do not replace an existing member. Default: N If you specify REPLACE=Y for a member, you cannot use the database file containing that member for any other operation during the replace operation.
RMTLOCNAME	Location Name	Name of the remote location with which you are communicating. Length: 8 characters Type: Input, required Valid values: Remote location name in a varied-on device description.
PASSWORD	PASSWORD	Password for signing on the remote system. Length: 10 characters Type: Input. This field is required only if the remote system has password security active. Valid values: Password

Figure E-2 (Page 3 of 3). Transferring Files to and from an AS/400 System

Parameter	Value	Description
RTNCODE	Character	<p>This field contains the return code. FTS returns this value to the application program to indicate the result of the transfer.</p> <p>Length: 1 character</p> <p>Type: Output</p> <p>Valid values:</p> <ul style="list-style-type: none"> • 0—Normal completion. • 1—An error was detected at the local system. • 2—An error was detected at the remote system. <p>For return codes 1 and 2, the specific error is sent to the job log of both systems, and the message-id is returned to the user in the <i>message-id</i> field. (See "File Transfer Support Messages" on page E-29 for more information.)</p>
MESSAGE-ID	Character	<p>This field contains the message-id for the specific error if the value returned in the RTNCODE field is 1 or 2 (indicating an error).</p> <p>Length: 8 characters</p> <p>Type: Output</p> <p>Valid values: Any message-id listed in the message section (See "File Transfer Support Messages" on page E-29 for more information.)</p>

AS/400 System Sending to System/36

Figure E-3 describes the parameters for sending from an AS/400 system to System/36.

Note: The TYPE parameter determines how data is stored on the System/36, therefore, it is important to specify the value of the TYPE

parameter correctly. If this parameter is not specified correctly, results may occur that cannot be predicted.

Figure E-3 (Page 1 of 3). AS/400 System Sending to System/36

Parameter	Value	Description
OPTION	Character	<p>File transfer option to perform.</p> <p>Length: 1 character</p> <p>Type: Input, required</p> <p>Valid values are as follows:</p> <ul style="list-style-type: none"> • S—Send • R—Retrieve. Refer to "AS/400 System Retrieving a File from System/36" on page E-9. This section describes only send.
FROMLIB	Library name	<p>Name of the library that contains the database file.</p> <p>Length: 10 characters</p> <p>Type: Input, required</p> <p>Valid values: Valid library name</p>
FROMFILE	File name	<p>Name of the database file that contains the member.</p> <p>Length: 10 characters</p> <p>Type: Input, required</p> <p>Valid values: Valid database file name</p>

Figure E-3 (Page 2 of 3). AS/400 System Sending to System/36

Parameter	Value	Description
FROMMBR	File member name	Name of the member. Length: 10 characters Type: Input, required Valid values: Valid member name
TYPE	File member type	This field tells System/36 how to store this member. Length: 6 characters Type: Input, required Valid values: SOURCE, LOAD, PROC (procedure), SUBR (subroutine), valid system date, or blanks. SOURCE, LOAD, PROC, or SUBR must be used if the member is to be stored as a library member. A date or blanks can be used if the member is to be stored as a file.
TOLIB	Target library name	Name of the receiving library to which the member is sent. If the member's attributes indicate that this is a System/36 file, this parameter must be left blank and the TOFILE parameter must be specified. Length: 10 characters (see note at the end of this figure) Type: Input, not required Valid values: System/36 library name, System/36 file name, and blanks Default: FROMLIB
TOFILE	Target file name	Name of the file. If the attributes of the member being sent indicate that this is a System/36 file, this parameter is used as the file name on System/36. Length: 10 characters (see note at the end of this figure) Type: Input, not required Valid values: System/36 file name and blanks Default: FROMFILE
TOMBR	Target member name	Name of the library member. If the attributes of the member being sent indicate that this is a System/36 file, this parameter is not used. Length: 10 characters (see note at the end of this figure) Type: Input, not required Valid values: System/36 library member name Default: FROMMBR When using TOMBR, consider the following: <ul style="list-style-type: none"> • If you are replacing a library member (REPLACE=Y), this is the name of the library member to replace at the target system. • If you are adding a new library member, this is the name to assign.
TODATE	Numeric	This field is used to change the date of a file sent to a System/36. Make sure that the system date format on the target system is the same as the format on your system. This field is not used if the member being sent will be used as a library member. Length: 6 characters Type: Input, not required Valid values: Numeric date

Figure E-3 (Page 3 of 3). AS/400 System Sending to System/36

Parameter	Value	Description
REPLACE	Character	<p>This field tells whether you want to replace the file or library member on the target system.</p> <p>Length: 1 character</p> <p>Type: Input, not required</p> <p>Valid values:</p> <ul style="list-style-type: none">• Y—Replace an existing file or library member on the target system.• N—Do not replace an existing file or library member. <p>Default: N</p> <p>If you specify REPLACE=Y for a library member, you cannot use the library containing that member for any other operation during the replace operation. If you specify REPLACE=Y for a file, the file cannot be used for any other operation during the replace operation.</p>
RMTLOCNAME	Location Name	<p>Name of the remote location with which you are communicating.</p> <p>Length: 8 characters</p> <p>Type: Input, required</p> <p>Valid values: Remote location name in a varied-on device description.</p>
PASSWORD	Password	<p>Password for signing on the remote system.</p> <p>Length: 10 characters. The largest password System/36 accepts is 4 characters.</p> <p>Type: Input. This field is required only if the remote system has password security active.</p> <p>Valid values: Password</p>
RTNCODE	Character	<p>This field contains the return code. FTS returns this value to the application program to indicate the result of the transfer.</p> <p>Length: 1 character</p> <p>Type: Output</p> <p>Valid values:</p> <ul style="list-style-type: none">• 0—Normal completion.• 1—An error was detected at the local system.• 2—An error was detected at the remote system. <p>For return codes 1 and 2, the specific error is logged to the history file of the System/36 and to the job log file of the AS/400 system. The message-id is returned to the user in the <i>message-id</i> field. (See “File Transfer Support Messages” on page E-29 for more information.)</p>
MESSAGE-ID	Character	<p>This field contains the message-id for the specific error if the value returned in the RTNCODE field is 1 or 2 (indicating an error).</p> <p>Length: 8 characters</p> <p>Type: Output</p> <p>Valid values: Any message-id listed in the message section. (See “File Transfer Support Messages” on page E-29 for more information.)</p>

Note: If a library name, file name, or member name is greater than 8 characters, System/36 uses the first 8 characters as the name.

AS/400 System Retrieving a File from System/36

Figure E-4 describes the parameters for retrieving a file from System/36.

Figure E-4 (Page 1 of 3). AS/400 System Retrieving a File from System/36

Parameter	Value	Description
OPTION	Character	File transfer option to perform. Length: 1 character Type: Input, required Valid values are as follows: <ul style="list-style-type: none"> • S—Send. Refer to “AS/400 System Sending to System/36” on page E-6. This section describes only retrieve. • R—Retrieve a file from the remote System/36.
FROMLIB	Blanks	Not needed in retrieving a file from System/36. Length: 10 characters Type: Not applicable Valid values: Blanks
FROMFILE	File name	Name of the System/36 file. Length: 10 characters (see note at the end of this figure) Type: Input, required Valid values: System/36 file name
FROMMBR	Blanks	Not needed in retrieving a file from System/36. Length: 10 characters Type: Not applicable Valid values: Blanks
TYPE	File type	This field contains the date of the file to retrieve. Make sure that the system date format on the target system is the same as the format on your system. Length: 6 characters Type: Input, not required Valid values: Numeric date Default: If no date is given, the most recent file is retrieved.
TOLIB	Target library name	The name of the receiving library. Length: 10 characters Type: Input, not required Valid values: Library name Default: The System/36 environment default library name.
TOFILE	Target file name	Name of the receiving database file. Length: 10 characters Type: Input, not required Valid values: File name Default: FROMFILE

Parameter	Value	Description
TOMBR	Target file member name	<p>Name of the receiving member.</p> <p>Length: 10 characters</p> <p>Type: Input, not required</p> <p>Valid values: Member name</p> <p>Default: If the name is not given, the member name is the date given in TODATE or the creation date with an <i>M</i> added as the first character.</p> <p>When using TOMBR, consider the following:</p> <ul style="list-style-type: none"> • If you are replacing a member (REPLACE=Y), TOMBR is the name of the member to replace at the target system. • If you are adding a new member, TOMBR is the name to assign.
TODATE	Numeric	<p>This field gives a different date to a file received from a System/36. Make sure that the system date format on the target system is the same as the format on your system.</p> <p>Length: 6 characters</p> <p>Type: Input, not required</p> <p>Valid values: Numeric date</p>
REPLACE	Character	<p>This field tells whether you want to replace the member.</p> <p>Length: 1 character</p> <p>Type: Input, not required</p> <p>Valid values:</p> <ul style="list-style-type: none"> • Y—Replace an existing member. • N—Do not replace an existing member. <p>Default: N</p> <p>If you specify REPLACE=Y for a member, you cannot use the database file containing that member for any other operation during the replace operation.</p>
RMTLOCNAME	Location Name	<p>Name of the remote location with which you are communicating.</p> <p>Length: 8 characters</p> <p>Type: Input, required</p> <p>Valid values: Remote location name in a varied-on device description.</p>
PASSWORD	Password	<p>Password for signing on the remote system.</p> <p>Length: 10 characters. The largest password System/36 accepts is 4 characters.</p> <p>Type: Input. This field is required only if the remote system has password security active.</p> <p>Valid values: Password</p>

Figure E-4 (Page 3 of 3). AS/400 System Retrieving a File from System/36

Parameter	Value	Description
RTNCODE	Character	<p>This field contains the return code. FTS returns this value to the application program to indicate the result of the transfer.</p> <p>Length: 1 character</p> <p>Type: Output</p> <p>Valid values:</p> <ul style="list-style-type: none"> • 0—Normal completion. • 1—An error was detected at the local system. • 2—An error was detected at the remote system. <p>For return codes 1 and 2, the specific error is logged to the history file of the System/36 and to the job log file of the AS/400 system. The message-id is returned to the user in the <i>message-id</i> field. (See “File Transfer Support Messages” on page E-29 for more information.)</p>
MESSAGE-ID	Character	<p>This field contains the message-id for the specific error if the value returned in the RTNCODE field is 1 or 2 (indicating an error).</p> <p>Length: 8 characters</p> <p>Type: Output</p> <p>Valid values: Any message-id listed in the message section. (See “File Transfer Support Messages” on page E-29 for more information.)</p>

Note: If a library name, file name, or member name is greater than 8 characters, System/36 uses the first 8 characters as the name.

Retrieving a Library Member from System/36

Figure E-5 describes the parameters for retrieving a library member from System/36.

Figure E-5 (Page 1 of 3). Retrieving a Library Member from System/36

Parameter	Value	Description
OPTION	Character	<p>File transfer option to perform.</p> <p>Length: 1 character</p> <p>Type: Input, required</p> <p>Valid values are as follows:</p> <ul style="list-style-type: none"> • S—Send. Refer to “AS/400 System Sending to System/36” on page E-6. This section describes only retrieve. • R—Retrieve a library member.
FROMLIB	Library name	<p>Name of the library in which library member resides.</p> <p>Length: 10 characters (see note)</p> <p>Type: Input, required</p> <p>Valid values: System/36 library name</p>
FROMFILE	Blanks	<p>Not needed in retrieving a library member from System/36.</p> <p>Length: 10 characters</p> <p>Type: Not applicable</p> <p>Valid values: Blanks</p>

Figure E-5 (Page 2 of 3). Retrieving a Library Member from System/36

Parameter	Value	Description
FROMMBR	Library member name	Name of the library member. Length: 10 characters (see note) Type: Input, required Valid values: System/36 member name
TYPE	Library member type	This field tells System/36 the type of member to retrieve. Length: 6 characters Type: Input, required Valid values: SOURCE, LOAD, PROC (procedure), SUBR (subroutine)
TOLIB	Library name	Name of the receiving library to which members are sent. Length: 10 characters Type: Input, not required Valid values: Library name Default: FROMLIB
TOFILE	Target file name	Name of the file. Length: 10 characters Type: Input, not required Valid values: Database file name Default: If no value is given, the default name is determined by the type of the member, as follows: <ul style="list-style-type: none"> • Source—QS36SRC • Load—QS36LOD • Procedure—QS36PRC • Subroutine—QS36SBR
TOMBR	Target member name	Name of the member at the target system. Length: 10 characters Type: Input, not required Valid values: Member name Default: FROMMBR When you use TOMBR, consider the following: <ul style="list-style-type: none"> • If you are replacing a member (REPLACE=Y), TOMBR is the name of the member to replace at the target system. • If you are adding a new member, TOMBR is the name to assign.
TODATE	Blanks	Not needed in retrieving a library member from System/36. Length: 6 characters Type: Input, not applicable Valid values: Blanks

Figure E-5 (Page 3 of 3). Retrieving a Library Member from System/36

Parameter	Value	Description
REPLACE	Character	<p>This field tells whether you want to replace the member on the target system.</p> <p>Length: 1 character</p> <p>Type: Input, not required</p> <p>Valid values:</p> <ul style="list-style-type: none"> • Y—Replace an existing member. • N—Do not replace an existing member. <p>Default: N</p> <p>If you specify REPLACE=Y for a member, you cannot use the file containing that member for any other operation during the replace operation.</p>
RMTLOCNAME	Location Name	<p>Name of the remote location with which you are communicating.</p> <p>Length: 8 characters</p> <p>Type: Input, required</p> <p>Valid values: Remote location name in a varied-on device description.</p>
PASSWORD	PASSWORD	<p>Password for signing on the remote system.</p> <p>Length: 10 characters. The largest password System/36 accepts is 4 characters.</p> <p>Type: Input. This field is required only if the remote system has password security active.</p> <p>Valid values: Password</p>
RTNCODE	Character	<p>This field contains the return code. FTS returns this value to the application program to indicate the result of the transfer.</p> <p>Length: 1 character</p> <p>Type: Output</p> <p>Valid values:</p> <ul style="list-style-type: none"> • 0—Normal completion. • 1—An error was detected at the local system. • 2—An error was detected at the remote system. <p>For return codes 1 and 2, the specific error is logged to the history log of the System/36 and to the job log file of the AS/400 system. The message-id is returned to the user in the <i>message-id</i> field. (See "File Transfer Support Messages" on page E-29 for more information.)</p>
MESSAGE-ID	Character	<p>This field contains the message-id for the specific error if the value returned in the RTNCODE field is 1 or 2 (indicating an error).</p> <p>Length: 8 characters</p> <p>Type: Output</p> <p>Valid values: Any message-id listed in the message section. (See "File Transfer Support Messages" on page E-29 for more information.)</p>

Note: If a library name, file name, or member name is greater than 8 characters, System/36 uses the first 8 characters as the name.

Calling File Transfer Support for C/400 Programming Language

Figure E-6 is an example of a C/400 program that provides a data link between one AS/400 system and another AS/400 system. This program reads the file in which the parameters

are stored, calls the file transfer support program (QY2FTML), and prints a listing of the parameters, return code, and message number.

The parameters passed to the file transfer program are described in "File Transfer Parameters" on page E-4.

```

5738CB1 V2R1M0 910524 IBM SAA C/400 CEXAMPLES/CDRIVER RCHAS400 11/09/90 14:12:32 Page 1
***** P R O L O G *****
Program name . . . . . : CDRIVER
Library name . . . . . : CEXAMPLES
Source file . . . . . : QCSRC
Library name . . . . . : CEXAMPLES
Source member name . . . . . : CDRIVER
Text Description . . . . . : File transfer program
Compiler options . . . . . : *SOURCE *NOXREF *NOSHOWUSR *NOSHOWSYS *NOSHOWSKP *NOEXPMAC *NOAGR
                              *NOPPONLY *NODEBUG *GEN *NOSECLVL *PRINT *LOGMSG
Language level options . . . . . :
Source margins:
Left margin . . . . . : 1
Right margin . . . . . : 80
Sequence columns:
Left Column . . . . . :
Right Column . . . . . :
Define name . . . . . :
Generation options . . . . . : *NOLIST *NOXREF *GEN *NOATR *NODUMP *NOOPTIMIZE *NOALWBND
                              *NOANNO
Print file . . . . . : QSYSVRT
Library name . . . . . : *LIBL
Message flagging level . . . . . : 0
Compiler message:
Message limit . . . . . : *NOMAX
Message limit severity . . . . . : 30
Replace program object . . . . . : *YES
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Target Release . . . . . : *CURRENT
Last change . . . . . : 90/11/09 14:07:57
Source description . . . . . : File transfer program
Compiler . . . . . : IBM SAA C/400 Compiler
5738CB1 V2R1M0 910524 IBM SAA C/400 CEXAMPLES/CDRIVER RCHAS400 11/09/90 14:12:32 Page 2
***** S O U R C E *****
Line STMT                      SEQNBR  INCNO
1  |#pragma linkage(QY2FTML,OS)    |          |
2  |                               |          |
3  |#define END 1                 |          |
4  |#define NOEND 0               |          |
5  |#define ERROR 1               |          |
6  |#define NOERROR 0             |          |
7  |#include <stdio.h>             |          |
8  |#include <recio.h>             |          |
9  |#include <stdlib.h>            |          |
10 |#include <stddef.h>            |          |
11 |#include <string.h>            |          |
12 |#include <xxfdbk.h>            |          |
13 |#include <xxasio.h>            |          |
14 |                               |          |
15 |/*-----*/                   |          |
16 |/* Define header structures.   |          |
17 |/*-----*/                   |          |
18 |                               |          |
19 |struct {                       |          |
20 |    char filler??(124??);      |          |
21 |} header_line_1 = {           |          |
22 |    "                               |          |
23 |    "FRMLIB "                  |          |
24 |    "FRMFIL "                  |          |
25 |    "FRMMBR "                  |          |

```

Figure E-6 (Part 1 of 7). C/400 Coding for File Transfer Support

26		"TYPE "		26
27		"TOLIB "		27
28		"TOLIB "		28
29		"TOMBR "		29
30		"TODATE "		30
31		"OPTION "		31
32		"REPL "		32
33		"RMTLOC "		33
34		"RCODE "		34
35		"MSGNUM "		35
36		});		36
37				37
38		struct {		38
39		char filler??(124??);		39
40		} header_line_2 = {		40
41		"		41
42		"_____"		42
43		"_____"		43
44		"_____"		44
45		"_____"		45
46		"_____"		46
47		"_____"		47
48		"_____"		48
49		"_____"		49
50		"_____"		50
51		"_____"		51
52		"_____"		52

5738CB1 V2R1M0 910524 IBM SAA C/400 CEX*MPLES/CDRIVER RCHAS400 11/09/90 14:12:32 Page 3

Line	STMT	SEQNBR	INCNO
53		"_____"	53
54		"_____"	54
55		});	55
56			56
57		/*-----*/	57
58		/* Define data file structure that contains the values to be assigned */	58
59		/* to parameters passed on call to program QY2FMTL. */	59
60		/*-----*/	60
61			61
62		struct {	62
63		char rec_num??(3??);	63
64		char option;	64
65		char repl;	65
66		char filler1??(4??);	66
67		char frmllib??(10??);	67
68		char frmfil??(10??);	68
69		char frmubr??(10??);	69
70		char typ??(6??);	70
71		char filler2??(4??);	71
72		char tolib??(10??);	72
73		char tofil??(10??);	73
74		char tombr??(10??);	74
75		char todate??(6??);	75
76		char filler3??(4??);	76
77		char rmtloc??(8??);	77
78		char passwd??(10??);	78
79		char rcode;	79
80		char msgnum??(8??);	80
81		} call_rec;	81
82			82
83		/*-----*/	83
84		/* Define structures of parameters to be passed on call to QY2FMTL. */	84
85		/* The type definition is needed for prototyping the call to QY2FMTL. */	85
86		/*-----*/	86
87			87
88		typedef struct {	88
89		char option;	89
90		} option;	90
91			91
92		typedef struct {	92
93		char frmllib??(10??);	93
94		} lib;	94
95			95
96		typedef struct {	96
97		char frmfil??(10??);	97
98		} file;	98
99			99
100		typedef struct {	100
101		char frmubr??(10??);	101
102		} file_member;	102

Figure E-6 (Part 2 of 7). C/400 Coding for File Transfer Support

Line	STMT	SEQNBR	INCNO
103		103	
104	typedef struct {	104	
105	char typ??(6??);	105	
106	} type;	106	
5738CB1 V2R1M0 910524 IBM SAA C/400	CEXAMPLES/CDRIVER	RCHAS400	11/09/90 14:12:32 Page 4
107	*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9.....	107	
108	typedef struct {	108	
109	char tolib??(10??);	109	
110	} tgt_lib;	110	
111		111	
112	typedef struct {	112	
113	char tofil??(10??);	113	
114	} tgt_file;	114	
115		115	
116	typedef struct {	116	
117	char toibr??(10??);	117	
118	} tgt_member;	118	
119		119	
120	typedef struct {	120	
121	char todate??(6??);	121	
122	} tgt_file_date;	122	
123		123	
124	typedef struct {	124	
125	char repl;	125	
126	} replace_member;	126	
127		127	
128	typedef struct {	128	
129	char rmtloc??(8??);	129	
130	} rmt_loc;	130	
131		131	
132	typedef struct {	132	
133	char passwd??(10??);	133	
134	} pword;	134	
135		135	
136	typedef struct {	136	
137	char rcode;	137	
138	} ret_code;	138	
139		139	
140	typedef struct {	140	
141	char msgnum??(8??);	141	
142	} msg_num;	142	
143		143	
144	/*-----*/	144	
145	/* Define print file structure. */	145	
146	/*-----*/	146	
147		147	
148	struct {	148	
149	char prec_num??(3??);	149	
150	char filler1??(2??);	150	
151	char pfrmfil??(10??);	151	
152	char filler2;	152	
153	char pfrmfil??(10??);	153	
154	char filler3;	154	
155	char pfrmbr??(10??);	155	
156	char filler4;	156	
157	char ptyp??(6??);	157	
158	char filler5??(3??);	158	
159	char ptolib??(10??);	159	
160	char filler6;	160	
5738CB1 V2R1M0 910524 IBM SAA C/400	CEXAMPLES/CDRIVER	RCHAS400	11/09/90 14:12:32 Page 5
161	*...+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9.....	161	
162	char ptofil??(10??);	162	
163	char ptombr??(10??);	163	
164	char filler8;	164	
165	char ptodate??(6??);	165	
166	char filler9??(5??);	166	
167	char poption;	167	
168	char filler10??(5??);	168	
169	char prepl;	169	
170	char filler11??(3??);	170	
171	char prmtloc??(8??);	171	
172	char filler12??(3??);	172	
173	char prcode;	173	
174	char filler13??(2??);	174	
175	char pmsgnum??(8??);	175	
176	} print_rec;	176	

Figure E-6 (Part 3 of 7). C/400 Coding for File Transfer Support

177	/*-----*/	177
178	/* Declare structures to use as parameters on call to QY2FTML using the */	178
179	/* type definitions already defined. */	179
180	/*-----*/	180
181	/*-----*/	181
182		182
183	option option_parm;	183
184	lib lib_parm;	184
185	file file_parm;	185
186	file_member file_member_parm;	186
187	type type_parm;	187
188	tgt_lib tgt_lib_parm;	188
189	tgt_file tgt_file_parm;	189
190	tgt_member tgt_member_parm;	190
191	tgt_file_date tgt_file_date_parm;	191
192	replace_member replace_member_parm;	192
193	rmt_loc rmt_loc_parm;	193
194	pword pword_parm;	194
195	ret_code ret_code_parm;	195
196	msg_num msg_num_parm;	196
197		197
198	char op_name??(???); /* Current operation */	198
199		199
200	extern void QY2FTML(option *, lib *, file *, file_member *, type *,	200
201	tgt_lib *, tgt_file *, tgt_member *, tgt_file_date *,	201
202	replace_member *, rmt_loc *, pword *, ret_code *,	202
203	msg_num *);	203
204	int print_header(_RFILE *);	204
205	int file_trans(_RFILE *);	205
206	void init_parms(void);	206
207	int print_parms(_RFILE *);	207
208	int pos_ret_code_printf(_RFILE *);	208
209	void print_file_error(void);	209
210	void close_files(_RFILE *, _RFILE *);	210
211	void get_access_to_fb(_RFILE *);	211
212		212
213	XXIOFB_T *comm_fdbk_ptr; /* Ptr to common I/O feedback */	213
214	XXIOFB_PRT_T *prt_fdbk; /* Ptr to printer I/O feedback */	214
5738CB1 V2R1M0 910524 IBM SAA C/400	CEXAMPLES/CDRIVER	RCHAS400
Line	STMT	SEQNBR
215	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9.....	215
216	main()	216
217	{	217
218	_RFILE *dtafpnr; /* Pointer to the database file */	218
219	_RFILE *prtfpnr; /* Pointer to the printer file */	219
220		220
221	1 if ((dtafpnr = _Ropen("FTTEST", "rr riofb=y")) == NULL) {	221
222	2 printf("\nUNEXPECTED ERROR FOR DATA FILE.\n");	222
223	3 exit(ERROR);	223
224	4 }	224
225	5 if ((prtfpnr = _Ropen("QSYSPRT", "wr lrecl=124 riofb=y")) == NULL) {	225
226	6 _Rclose(dtafpnr);	226
227	7 print_file_error();	227
228	8 exit(ERROR);	228
229	9 }	229
230	10 if (print_header(prtfpnr) == NOERROR) {	230
231	11 while (1) {	231
232	12 if (file_trans(dtafpnr) == END)	232
233	13 break;	233
234	14 else	234
235	15 if (print_parms(prtfpnr) == ERROR) {	235
236	print_file_error();	236
237	close_files(dtafpnr, prtfpnr);	237
238	exit(ERROR);	238
239	}	239
240	}	240
241	}	241
242	else {	242
243	print_file_error();	243
244	close_files(dtafpnr, prtfpnr);	244
245	exit(ERROR);	245
246	}	246
247	}	247
248		248
249		249
250	/*-----*/	250
251	/* The routine prints a header to the print file. */	251
252	/*-----*/	252
253		253

Figure E-6 (Part 4 of 7). C/400 Coding for File Transfer Support

254	print_header(_RFILE *prtfptr)	254
255	{	255
256	1 strcpy(op_name, "WRITE");	256
257	2 _Rwrite(prtfptr, &header_line_1, sizeof(header_line_1));	257
258	3 if (pos_ret_code_printf(prtfptr) == NOERROR) {	258
259	4 _Rwrite(prtfptr, &header_line_2, sizeof(header_line_2));	259
260	5 return(pos_ret_code_printf(prtfptr));	260
261	}	261
262	else	262
263	6 return(ERROR);	263
264	}	264
265	}	265
266	}	266
267	/*-----*/	267
268	/* The routine gets parameters from the data file and calls QY2FTML. */	268
5738CB1 V2R1M0 910524 IBM SAA C/400 CEXAMPLES/CDRIVER RCHAS400 11/09/90 14:12:32		Page 7
Line	STMT	SEQNBR
	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9.....	
269	/*-----*/	269
270		270
271	file_trans(_RFILE *dtafptr)	271
272	{	272
273	_RIOFB_T *rio_fdbk;	273
274		274
275	1 strcpy(op_name, "READ");	275
276	2 rio_fdbk = _Rreadn(dtafptr, &call_rec, sizeof(call_rec), _DFT);	276
277	3 if (rio_fdbk->num_bytes != -1) { /* Record found */	277
278	4 init_parms();	278
279	QY2FTML(&option_parm, &lib_parm, &file_parm, &file_member_parm,	279
280	&type_parm, &tgt_lib_parm, &tgt_file_parm, &tgt_member_parm,	280
281	&tgt_file_date_parm, &replace_member_parm, &rmt_loc_parm,	281
282	5 &pwd_parm, &ret_code_parm, &msg_num_parm);	282
283	6 return(NOEND);	283
284	}	284
285	else	285
286	7 return(END);	286
287	}	287
288	}	288
289	}	289
290	/*-----*/	290
291	/* This routine initializes the parameters for the call to QY2FTML. */	291
292	/* Parameters passed in C/400 language must be a structure type, so */	292
293	/* that subfields of the structure call_rec may not be sent directly. */	293
294	/*-----*/	294
295		295
296	void init_parms()	296
297	{	297
298	1 option_parm.option = call_rec.option;	298
299	2 strcpy(lib_parm.frmlib, call_rec.frmlib);	299
300	3 strcpy(file_parm.frmfil, call_rec.frmfil);	300
301	4 strcpy(file_member_parm.frmnbr, call_rec.frmnbr);	301
302	5 strcpy(type_parm.typ, call_rec.typ);	302
303	6 strcpy(tgt_lib_parm.tolib, call_rec.tolib);	303
304	7 strcpy(tgt_file_parm.tofil, call_rec.tofil);	304
305	8 strcpy(tgt_member_parm.tombr, call_rec.tombr);	305
306	9 strcpy(tgt_file_date_parm.todate, call_rec.todate);	306
307	10 replace_member_parm.repl = call_rec.repl;	307
308	11 strcpy(rmt_loc_parm.rmtloc, call_rec.rmtloc);	308
309	12 strcpy(pwd_parm.passwd, call_rec.passwd);	309
310	13 ret_code_parm.rcode = ' ';	310
311	14 strcpy(msg_num_parm.msgnum, " ");	311
312	}	312
313	}	313
314	}	314
315	/*-----*/	315
316	/* This routine prints the parameters passed to QY2FTML after the call. */	316
317	/*-----*/	317
318		318
319	print_parms(_RFILE *prtfptr)	319
320	{	320
321	1 strncpy(print_rec.prec_num, call_rec.rec_num, 3);	321
322	2 print_rec.poption = option_parm.option;	322
5738CB1 V2R1M0 910524 IBM SAA C/400 CEXAMPLES/CDRIVER RCHAS400 11/09/90 14:12:32		Page 8
Line	STMT	SEQNBR
	*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9.....	
323	3 print_rec.prepl = replace_member_parm.repl;	323
324	4 strncpy(print_rec.pfrmlib, lib_parm.frmlib, 10);	324
325	5 strncpy(print_rec.pfrmfil, file_parm.frmfil, 10);	325
326	6 strncpy(print_rec.pfrmmbr, file_member_parm.frmnbr, 10);	326
327	7 strncpy(print_rec.ptyp, type_parm.typ, 6);	327

Figure E-6 (Part 5 of 7). C/400 Coding for File Transfer Support

```

328 8 | strncpy(print_rec.ptolib, tgt_lib_parm.tolib, 10); | 328
329 9 | strncpy(print_rec.ptofil, tgt_file_parm.tofil, 10); | 329
330 10 | strncpy(print_rec.ptombr, tgt_member_parm.tombr, 10); | 330
331 11 | strncpy(print_rec.ptodate, tgt_file_date_parm.todate, 6); | 331
332 12 | strncpy(print_rec.prtloc, rmt_loc_parm.rmtloc, 8); | 332
333 13 | print_rec.prcode = ret_code_parm.rcode; | 333
334 14 | strncpy(print_rec.pmsgnum, msg_num_parm.msgnum, 8); | 334
335 15 | strcpy(op_name, "WRITE"); | 335
336 16 | _Rwrite(prtfptr, &print_rec, sizeof(print_rec)); | 336
337 17 | return(pos_ret_code_printf(prtfptr)); | 337
338 | } | 338
339 | | 339
340 | | 340
341 | /*-----*/ | 341
342 | /* This routine checks to see if the last operation on the print file */ | 342
343 | /* was successful. */ | 343
344 | /*-----*/ | 344
345 | | 345
346 | pos_ret_code_printf(_RFILE *prtfptr) | 346
347 | { | 347
348 | 1 | get_access_to_fb(prtfptr); | 348
349 | 2 | if (memcmp(prt_fdbk->major_ret_code, "00", 2) == NOERROR) | 349
350 | 3 | return(NOERROR); | 350
351 | | else | 351
352 | 4 | return(ERROR); | 352
353 | | } | 353
354 | | | 354
355 | | | 355
356 | /*-----*/ | 356
357 | /* This routine prints an error message to the display. */ | 357
358 | /*-----*/ | 358
359 | | 359
360 | void print_file_error() | 360
361 | { | 361
362 | 1 | printf("\nUNEXPECTED ERROR ON %s FOR PRINT FILE\n", op_name); | 362
363 | | } | 363
364 | | | 364
365 | | | 365
366 | /*-----*/ | 366
367 | /* Close the data and print files. */ | 367
368 | /*-----*/ | 368
369 | | 369
370 | void close_files(_RFILE *dtafptr, _RFILE *prtfptr) | 370
371 | { | 371
372 | 1 | _Rclose(dtafptr); | 372
373 | 2 | _Rclose(prtfptr); | 373
374 | | } | 374
375 | | | 375
376 | | | 376
5738CB1 V2R1M0 910524 IBM SAA C/400 CEXAMPLES/CDRIVER RCHAS400 11/09/90 14:12:32 Page 9
Line STMT SEQNBR INCNO
377 | /*-----*/ | 377
378 | /* Get access to the printer file feedback area. */ | 378
379 | /*-----*/ | 379
380 | | 380
381 | void get_access_to_fb(_RFILE *prtfptr) | 381
382 | { | 382
383 | 1 | comm_fdbk_prt = Riofbk(prtfptr); | 383
384 | | prt_fdbk = (XXIOFB_PRT_T *) ((char *)comm_fdbk_prt + | 384
385 | 2 | comm_fdbk_prt->file_dep_fb_offset); | 385
386 | | } | 386
***** END OF SOURCE *****
5738CB1 V2R1M0 910524 IBM SAA C/400 CEXAMPLES/CDRIVER RCHAS400 11/09/90 14:12:32 Page 10
***** INCLUDES *****
INCNO Include Name Last change Actual Include Name
1 stdio.h 90/10/15 15:51:05 QCC/H/STDIO
2 stddef.h 90/10/15 15:51:04 QCC/H/STDDEF
3 errno.h 90/10/15 15:51:00 QCC/H/ERRNO
4 signal.h 90/10/15 15:51:03 QCC/H/SIGNAL
5 ctype.h 90/10/15 15:50:59 QCC/H/CTYPE
6 stdarg.h 90/10/15 15:51:04 QCC/H/STDARG
7 recio.h 90/10/15 15:51:02 QCC/H/RECIO
8 xxasio.h 90/10/15 15:51:08 QCC/H/XXASIO
9 xxfdbk.h 90/10/15 15:51:11 QCC/H/XXFDBK
10 stdlib.h 90/10/15 15:51:06 QCC/H/STDLIB
11 stddef.h 90/10/15 15:51:04 QCC/H/STDDEF
12 string.h 90/10/15 15:51:07 QCC/H/STRING
13 xxfdbk.h 90/10/15 15:51:11 QCC/H/XXFDBK
14 xxasio.h 90/10/15 15:51:08 QCC/H/XXASIO

```

Figure E-6 (Part 6 of 7). C/400 Coding for File Transfer Support

```

          ***** END OF INCLUDES *****
5738CB1 V2R1M0 910524 IBM SAA C/400 CEXAMPLES/CDRIVER RCHAS400 11/09/90 14:12:32 Page 11
          ***** MESSAGE SUMMARY *****
          Total      Info(0-4)      Warning(5-19)      Error(20-29)      Severe(30-39)      Terminal(40-99)
           0          0              0                  0                  0                  0
          ***** END OF MESSAGE SUMMARY *****

```

```

5738CB1 V2R1M0 910524 IBM SAA C/400 CEXAMPLES/CDRIVER RCHAS400 11/09/90 14:12:32 Page 12
ROUTINE      BLOCK NUMBER SCOPE TYPE
<MAIN>         2      LOCAL MAIN-PROGRAM
__reads        119     LOCAL PROCEDURE
__rwrite       122     LOCAL PROCEDURE
__memcmp       170     LOCAL PROCEDURE
__strcpy       185     LOCAL PROCEDURE
__strncpy      196     LOCAL PROCEDURE
print_header   227     ENTRY  PROCEDURE
file_trans     228     ENTRY  PROCEDURE
init_parms     229     ENTRY  PROCEDURE
print_parms    230     ENTRY  PROCEDURE
pos_ret_code_printf
                231     ENTRY  PROCEDURE
print_file_error
                232     ENTRY  PROCEDURE
close_files    233     ENTRY  PROCEDURE
get_access_to_fb
                234     ENTRY  PROCEDURE
main           237     ENTRY  PROCEDURE
Program CDRIVER was created in library CEXAMPLES.
          ***** END OF COMPILATION *****

```

Figure E-6 (Part 7 of 7). C/400 Coding for File Transfer Support

Calling File Transfer Support for COBOL/400 Programming Language

Figure E-7 is an example of a COBOL/400 program that provides a data link between one AS/400 system and another AS/400 system. This program reads the file in which the parameters

are stored, calls the file transfer support program (QY2FTML), and prints a listing of the parameters, return code, and message number.

The parameters passed to the file transfer program are described in "File Transfer Parameters" on page E-4.

```

5738CB1 V2R1M0 910524          IBM SAA COBOL/400          KPSLIB/COBDRIVER    RCHAS209 12/21/90 07:19:14    Page 1
Program . . . . . : COBDRIVER
Library . . . . . : KPSLIB
Source file . . . . . : QCBLSRC
Library . . . . . : KPSLIB
Source member . . . . . : COBDRIVER    05/05/89 09:23:28
Generation severity level . . . . . : 29
Text 'description' . . . . . : *SAME
Source listing options . . . . . : *NONE
Generation options . . . . . : *NONE
Print file . . . . . : QSYSVRT
Library . . . . . : *LIBL
FIPS flagging . . . . . : *NOFIPS *NOSEG *NODEB *NOOBSOLETE
SAA flagging . . . . . : *NOFLAG
Flagging severity . . . . . : 0
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Compiler . . . . . : IBM SAA COBOL/400
5738CB1 V2R1M0 910524          AS/400 COBOL Source      KPSLIB/COBDRIVER    RCHAS209 12/21/90 07:19:14    Page 2
STMT SEQNBR -A 1 B..+....2....+....3....+....4....+....5....+....6....+....7..IDENTFCN S COPYNAME  CHG DATE
1 000100 IDENTIFICATION DIVISION.                                     11/04/87
2 000200 PROGRAM-ID. QY2FTML.                                         11/04/87
3 000300 AUTHOR. EAPOE.                                               11/04/87
  000400*                                                                11/04/87
4 000500 ENVIRONMENT DIVISION.                                        11/04/87
5 000600 CONFIGURATION SECTION.                                       11/04/87
6 000700 SOURCE-COMPUTER. IBM-AS400.                                   05/04/89
7 000800 OBJECT-COMPUTER. IBM-AS400.                                   05/04/89
8 000900 INPUT-OUTPUT SECTION.                                         11/04/87
9 001000 FILE-CONTROL.                                                11/04/87
10 001100 SELECT SEQ-FILE ASSIGN TO DISK-FTTEST                       01/29/88
11 001200 ORGANIZATION IS SEQUENTIAL                                  11/04/87
12 001300 FILE STATUS IS SEQ-FILE-STATUS.                             05/04/89
  001400*                                                                11/04/87
13 001500 SELECT SYSVRT ASSIGN TO PRINTER-QSYSVRT,                    01/29/88
14 001600 ORGANIZATION IS SEQUENTIAL                                  01/29/88
15 001700 ACCESS IS SEQUENTIAL                                        05/04/89
16 001800 FILE STATUS IS PRINT-FILE-STATUS.                           05/04/89
  001900*                                                                01/29/88
17 002000 DATA DIVISION.                                             11/04/87
18 002100 FILE SECTION.                                               11/04/87
19 002200 FD SEQ-FILE LABEL RECORDS ARE STANDARD.                    11/04/87
20 002300 01 SEQ-FILE-REC.                                             11/04/87
21 002400 02 FILLER PIC X(101).                                        11/04/87
  002500*                                                                01/29/88
22 002600 FD SYSVRT                                                    01/29/88
23 002700 LABEL RECORDS ARE OMITTED                                   02/01/88
24 002800 LINAGE IS 80 LINES.                                         02/01/88
  002900*                                                                02/02/88
25 003000 01 PRINT-FILE-REC.                                          11/04/87
26 003100 02 FILLER PIC X(132).                                        11/04/87
  003200*                                                                02/02/88
27 003300 WORKING-STORAGE SECTION.                                    11/04/87
28 003400 77 SEQ-FILE-STATUS PIC X(2).                                01/29/88
29 003500 77 PRINT-FILE-STATUS PIC X(2).                              01/29/88
30 003600 77 OP-NAME PIC X(7).                                        11/04/87
31 003700 77 ERRORFLAG PIC X VALUE SPACES.                           01/29/88
32 003800 77 EOF PIC X VALUE SPACES.                                  01/29/88
33 003900 77 EOF-FLAG PIC X VALUE "1".                                01/29/88
  004000*                                                                02/02/88
34 004100 01 ERRORFLAG PIC X VALUE SPACES.                           01/29/88
35 004200 88 ERROR-OCCURED VALUE "1".                                01/29/88
  004300*                                                                02/02/88

```

Figure E-7 (Part 1 of 4). COBOL/400 Coding for File Transfer Support

```

36 004400 01 HEADER-LINE-1.                                02/01/88
37 004500 03 FILLER PIC X(5) VALUE SPACES.                02/01/88
38 004600 03 FROM-LIBRARY PIC X(11) VALUE "FRMLIB " ".    02/01/89
39 004700 03 FROM-FILE PIC X(11) VALUE "FRMFIL " ".       02/01/89
40 004800 03 FROM-MEMBER PIC X(11) VALUE "FRMMBR " ".     02/01/89
41 004900 03 OBJ-TYPE PIC X(9) VALUE "TYPE " ".           02/01/89
42 005000 03 TO-LIBRARY PIC X(11) VALUE "TOLIB " ".       02/01/89
43 005100 03 TO-FILE PIC X(11) VALUE "TOFIL " ".          02/01/89
44 005200 03 TO-MEMBER PIC X(11) VALUE "TOMBR " ".        02/01/89
45 005300 03 TO-DATE PIC X(9) VALUE "TODATE " ".          02/01/89
46 005400 03 OPTN PIC X(7) VALUE "OPTION " ".              02/01/89
47 005500 03 REPLCE PIC X(5) VALUE "REPL " ".              02/01/89
5738CB1 V2R1M0 910524 AS/400 COBOL Source KPSLIB/COBDRIVER RCHAS209 12/21/90 07:19:14 Page 3
STMT SEQNBR -A 1 B. ....2.....3.....4.....5.....6.....7..IDENTFCN S COPYNAME CHG DATE
48 005600 03 REMOTE-LOCATION PIC X(9) VALUE "RMTLOC " ".    02/01/89
49 005700 03 RETURN-CODE PIC X(6) VALUE "RCODE " ".       02/01/89
50 005800 03 MESSAGE-NUMBER PIC X(7) VALUE "MSGNUM " ".    02/01/89
005900* 02/02/88
51 006000 01 HEADER-LINE-2.                                02/01/88
52 006100 03 FILLER PIC X(5) VALUE SPACES.                02/01/88
53 006200 03 FILLER PIC X(11) VALUE " " ".                02/01/89
54 006300 03 FILLER PIC X(11) VALUE " " ".                02/01/89
55 006400 03 FILLER PIC X(11) VALUE " " ".                02/01/89
56 006500 03 FILLER PIC X(9) VALUE " " ".                 02/01/89
57 006600 03 FILLER PIC X(11) VALUE " " ".                02/01/89
58 006700 03 FILLER PIC X(11) VALUE " " ".                02/01/89
59 006800 03 FILLER PIC X(11) VALUE " " ".                02/01/89
60 006900 03 FILLER PIC X(9) VALUE " " ".                 02/01/89
61 007000 03 FILLER PIC X(7) VALUE " " ".                 02/01/89
62 007100 03 FILLER PIC X(5) VALUE " " ".                 02/01/89
63 007200 03 FILLER PIC X(9) VALUE " " ".                 02/01/89
64 007300 03 FILLER PIC X(6) VALUE " " ".                 02/01/89
65 007400 03 FILLER PIC X(8) VALUE " " ".                 02/01/89
007500* 02/02/88
66 007600 01 CALL-REC. 11/04/87
007700* RECORD NUMBER 02/01/88
67 007800 02 REC-NUM PIC X(3). 02/01/88
007900* OPTION 02/01/88
68 008000 02 OPTION PIC X(1). 02/01/88
008100* REPLACE MEMBER 02/01/88
69 008200 02 REPL PIC X(1). 02/01/88
008300* BLANKS 02/01/88
70 008400 02 FILLER PIC X(4). 02/01/88
008500* LIBRARY NAME 02/01/88
71 008600 02 FRMLIB PIC X(10). 02/01/88
008700* FILE NAME 02/01/88
72 008800 02 FRMFIL PIC X(10). 02/01/88
008900* FILE MEMBER 02/01/88
73 009000 02 FRMMBR PIC X(10). 02/01/88
009100* TYPE 02/01/88
74 009200 02 TYP PIC X(6). 02/01/88
009300* BLANKS 02/01/88
75 009400 02 FILLER PIC X(4). 02/01/88
009500* TARGET LIBRARY NAME 11/04/87
76 009600 02 TOLIB PIC X(10). 02/01/88
009700* TARGET FILE/LIBRARY NAME 11/04/87
77 009800 02 TOFIL PIC X(10). 02/01/88
009900* TARGET FILE/LIBRARY MEMBER NAME 11/04/87
78 010000 02 TOMBR PIC X(10). 02/01/88
010100* TARGET FILE DATE 11/04/87
79 010200 02 TODATE PIC X(6). 02/01/88
010300* BLANKS 02/01/88
80 010400 02 FILLER PIC X(4). 02/01/88
010500* REMOTE LOCATION 02/01/88
81 010600 02 RMTLOC PIC X(8). 02/01/88
010700* PASSWORD 11/04/87
82 010800 02 PASSWD PIC X(10). 02/24/89
010900* RETURN CODE 11/04/87
83 011000 02 RCODE PIC X(1). 02/24/89
5738CB1 V2R1M0 910524 AS/400 COBOL Source KPSLIB/COBDRIVER RCHAS209 12/21/90 07:19:14 Page 4
STMT SEQNBR -A 1 B. ....2.....3.....4.....5.....6.....7..IDENTFCN S COPYNAME CHG DATE
011100* MESSAGE NUMBER 02/01/88
84 011200 02 MSGNUM PIC X(8). 11/04/87
011300* 02/02/88
85 011400 01 PRINT-REC. 02/01/88
011500* PRINT RECORD NUMBER 02/01/88
86 011600 02 PREC-NUM PIC X(3). 02/01/88
011700* BLANKS 02/01/88
87 011800 02 FILLER PIC X(2). 02/01/88
011900* PRINT LIBRARY NAME 02/01/88

```

Figure E-7 (Part 2 of 4). COBOL/400 Coding for File Transfer Support

88	012000	02	PFRMLIB	PIC X(10).					02/01/88
	012100*		BLANKS						02/02/88
89	012200	02	FILLER	PIC X(1).					02/02/88
	012300*		FILE NAME						02/02/88
90	012400	02	PFRMFIL	PIC X(10).					02/02/88
	012500*		BLANKS						02/02/88
91	012600	02	FILLER	PIC X(1).					02/02/88
	012700*		FILE MEMBER						02/02/88
92	012800	02	PFRMMBR	PIC X(10).					02/02/88
	012900*		BLANKS						02/02/88
93	013000	02	FILLER	PIC X(1).					02/02/88
	013100*		TYPE						02/02/88
94	013200	02	PTYP	PIC X(6).					02/02/88
	013300*		BLANKS						02/02/88
95	013400	02	FILLER	PIC X(3).					02/02/88
	013500*		TARGET LIBRARY NAME						02/02/88
96	013600	02	PTOLIB	PIC X(10).					02/02/88
	013700*		BLANKS						02/02/88
97	013800	02	FILLER	PIC X(1).					02/02/88
	013900*		TARGET FILE/LIBRARY NAME						02/02/88
98	014000	02	PTOFLI	PIC X(10).					02/02/88
	014100*		BLANKS						02/02/88
99	014200	02	FILLER	PIC X(1).					02/02/88
	014300*		TARGET FILE/LIBRARY MEMBER NAME						02/02/88
100	014400	02	PTOMBR	PIC X(10).					02/02/88
	014500*		BLANKS						02/02/88
101	014600	02	FILLER	PIC X(1).					02/02/88
	014700*		TARGET FILE DATE						02/02/88
102	014800	02	PTODATE	PIC X(6).					02/02/88
	014900*		BLANKS						02/02/88
103	015000	02	FILLER	PIC X(5).					02/02/88
	015100*		PRINT OPTION						02/01/88
104	015200	02	POPTION	PIC X(1).					02/01/88
	015300*		BLANKS						02/02/88
105	015400	02	FILLER	PIC X(5).					02/02/88
	015500*		PRINT REPLACE MEMBER						02/01/88
106	015600	02	PREPL	PIC X(1).					02/01/88
	015700*		BLANKS						02/02/88
107	015800	02	FILLER	PIC X(3).					02/02/88
	015900*		REMOTE LOCATION						02/01/88
108	016000	02	PRMTLOC	PIC X(8).					02/02/88
	016100*		BLANKS						02/02/88
109	016200	02	FILLER	PIC X(3).					02/02/88
	016300*		RETURN CODE						02/01/88
110	016400	02	PRCODE	PIC X(2).					02/02/88
	016500*		BLANKS						02/02/88
5738CB1	V2R1M0	910524	AS/400 COBOL Source	KPSLIB/COBDRIVER	RCHAS209	12/21/90	07:19:14	Page	5
STMT	SEQNBR	-A	1	B.	+....2....+....3....+....4....+....5....+....6....+....7..	IDENTFCN	S	COPYNAME	CHG DATE
111	016600	02	FILLER	PIC X(2).					02/02/88
	016700*		MESSAGE NUMBER						02/01/88
112	016800	02	PMSGNUM	PIC X(8).					02/02/88
	016900*								02/02/88
113	017000		PROCEDURE DIVISION.						11/04/87
	017100		DECLARATIVES.						11/04/87
	017200		*****						05/04/89
	017300*		SEQUENTIAL DECLARATIVE SECTION						05/04/89
	017400*								05/04/89
	017500		*****						05/04/89
	017600		I-0-ERROR-SEQ SECTION.						11/04/87
	017700		USE AFTER STANDARD ERROR PROCEDURE ON SEQ-FILE.						11/04/87
	017800		I-0-ERROR-PARA-SEQ.						11/04/87
114	017900		DISPLAY "UNEXPECTED ERROR ON ", OP-NAME, "FOR SEQ FILE".						05/04/89
115	018000		DISPLAY "FILE STATUS IS " SEQ-FILE-STATUS.						05/04/89
116	018100		SET ERROR-OCCURED TO TRUE.						11/04/87
	018200		*****						05/04/89
	018300*		PRINTER FILE DECLARATIVE SECTION						05/04/89
	018400*								05/04/89
	018500		*****						05/04/89
	018600		I-0-ERROR-PRINT SECTION.						11/04/87
	018700		USE AFTER STANDARD ERROR PROCEDURE ON SYSPRT.						01/29/88
	018800		I-0-ERROR-PARA-PRINT.						11/04/87
117	018900		DISPLAY "UNEXPECTED ERROR ON ", OP-NAME, " FOR PRINT FILE".						11/09/87
118	019000		DISPLAY "FILE STATUS IS ", PRINT-FILE-STATUS.						11/09/87
119	019100		SET ERROR-OCCURED TO TRUE.						11/04/87
	019200		END DECLARATIVES.						11/04/87
	019300*								02/02/88
	019400		MAIN-PROGRAM SECTION.						11/04/87
	019500		MAIN-PROCEDURE.						11/04/87
120	019600		PERFORM OPEN-FILES.						02/01/88
121	019700		PERFORM HDR-PRT.						02/01/88

Figure E-7 (Part 3 of 4). COBOL/400 Coding for File Transfer Support

```

122 019800    PERFORM READ-REC.                                05/04/89
123 019900    PERFORM FILE-TRANS UNTIL EOF = EOF-FLAG.    01/29/88
124 020000    PERFORM CLOSE-FILES.                       11/04/87
125 020100    STOP RUN.                                  11/04/87
020200*
020300 FILE-TRANS.                                     02/02/88
126 020400    MOVE SPACES TO MSGNUM.                     11/09/87
127 020500    MOVE SPACES TO RCODE.                      11/09/87
128 020600    CALL "QY2FTML" USING OPTION FRMLIB FRMFIL FRMMBR TYP TOLIB
020700        TOFIL TOMBR TODATE REPL RMTLOC PASSWD RCODE MSGNUM. 02/01/88
129 020800    PERFORM PRINT-PARAMETERS.                  11/09/87
130 020900    PERFORM READ-REC.                           05/04/89
021000*
021100 READ-REC.                                       05/04/89
131 021200    MOVE "READ" TO OP-NAME.                    05/04/89
132 021300    READ SEQ-FILE INTO CALL-REC                 05/04/89
133 021400        AT END MOVE EOF-FLAG TO EOF.            05/04/89
134 021500    IF ERROR-OCCURED GO TO ERROR-TERMINATION. 05/04/89
021600*
021700 PRINT-PARAMETERS.                               11/09/87
136 021800    MOVE REC-NUM TO PREC-NUM.                  02/02/88
137 021900    MOVE OPTION TO POPTION.                    02/02/88
138 022000    MOVE REPL TO PREPL.                        02/02/88
5738CB1 V2R1M0 910524 AS/400 COBOL Source KPSLIB/COBDRIVER RCHAS209 12/21/90 07:19:14 Page 6
STMT SEQNBR -A 1 B..+...2....+...3....+...4....+...5....+...6....+...7..IDENTFCN S COPYNAME CHG DATE
139 022100    MOVE FRMLIB TO PFRMLIB.                    02/02/88
140 022200    MOVE FRMFIL TO PFRMFIL.                    02/02/88
141 022300    MOVE FRMMBR TO PFRMMBR.                    02/02/88
142 022400    MOVE TYP TO PTYP.                          02/02/88
143 022500    MOVE TOLIB TO PTOLIB.                      02/02/88
144 022600    MOVE TOFIL TO PTOFIL.                      02/02/88
145 022700    MOVE TOMBR TO PTOMBR.                      02/02/88
146 022800    MOVE TODATE TO PTODATE.                    02/02/88
147 022900    MOVE RMTLOC TO PRMTLOC.                    02/02/88
148 023000    MOVE RCODE TO PRCODE.                      02/02/88
149 023100    MOVE MSGNUM TO PMSGNUM.                    02/02/88
150 023200    MOVE "WRITE" TO OP-NAME.                   02/01/88
151 023300    WRITE PRINT-FILE-REC FROM PRINT-REC.       02/02/88
152 023400    IF ERROR-OCCURED GO TO ERROR-TERMINATION. 11/09/87
023500*
023600 HDR-PRT.                                         02/02/88
154 023700    MOVE "WRITE" TO OP-NAME.                   02/01/88
155 023800    WRITE PRINT-FILE-REC FROM HEADER-LINE-1.   02/01/88
156 023900    WRITE PRINT-FILE-REC FROM HEADER-LINE-2.   02/01/88
157 024000    IF ERROR-OCCURED GO TO ERROR-TERMINATION. 02/01/88
024100*
024200 OPEN-FILES.                                       11/09/87
159 024300    MOVE "OPEN" TO OP-NAME.                    11/09/87
160 024400    OPEN I-O SEQ-FILE,                          11/09/87
024500        OUTPUT SYSPRT.                              01/29/88
161 024600    IF ERROR-OCCURED GO TO ERROR-TERMINATION. 11/09/87
024700*
024800 CLOSE-FILES.                                       11/09/87
163 024900    MOVE "CLOSE" TO OP-NAME.                   11/09/87
164 025000    CLOSE SEQ-FILE SYSPRT.                     01/29/88
165 025100    IF ERROR-OCCURED GO TO ERROR-TERMINATION. 11/09/87
025200*
025300 ERROR-TERMINATION.                               11/09/87
167 025400    DISPLAY "I-O ERROR OCCURED - PROCESS TERMINATION". 11/09/87
168 025500    STOP RUN.                                  11/09/87
***** END OF SOURCE *****
5738CB1 V2R1M0 910524 AS/400 COBOL Messages KPSLIB/COBDRIVER RCHAS209 12/21/90 07:19:14 Page 7
STMT
***** END OF MESSAGES *****
Message Summary
Total Info(0-4) Warning(5-19) Error(20-29) Severe(30-39) Terminal(40-99)
0 0 0 0 0 0
Source records read . . . . . : 255
Copy records read . . . . . : 0
Copy members processed . . . . . : 0
Sequence errors . . . . . : 0
Highest severity message issued . . : 0
LBL0901 00 Program COBDRIVER created in library KPSLIB.
***** END OF COMPILATION *****

```

Figure E-7 (Part 4 of 4). COBOL/400 Coding for File Transfer Support

Calling File Transfer Support for RPG/400 Programming Language

Figure E-8 is an example of an RPG/400 program that provides a data link between one AS/400 system and another AS/400 system. The program shown reads the file in which the

parameters are stored, calls the file transfer support program (QY2FTML), and prints a listing of the parameters, return code, and message number.

The parameters passed to the file transfer are described in "File Transfer Parameters" on page E-4.

```

5738RG1 V2R1M0 910524          IBM SAA RPG/400          KPSLIB/RPGDRIVER  12/21/90 07:32:18  Page    1
Compiler . . . . . : IBM SAA RPG/400
Command Options:
Program . . . . . : KPSLIB/RPGDRIVER
Source file . . . . . : KPSLIB/QRPGSRC
Source member . . . . . : RPGDRIVER
Source listing options . . . . . : *SOURCE *XREF *GEN *NODUMP *NOSECLVL
Generation options . . . . . : *NOLIST *NOXREF *NOATR *NODUMP *NOOPTIMIZE
Source listing indentation . . . . . : *NONE
SAA flagging . . . . . : *NOFLAG
Generation severity level . . . . . : 9
Print file . . . . . : *LIBL/QSYSVRT
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
User profile . . . . . : *USER
Authority . . . . . : *LIBCRTAUT
Text . . . . . : *SRCMBRTXT
Phase trace . . . . . : *NO
Intermediate text dump . . . . . : *NONE
Snap dump . . . . . : *NONE
Codelist . . . . . : *NONE
Ignore decimal data error . . . . . : *NO
Actual Program Source:
Member . . . . . : RPGDRIVER
File . . . . . : QRPGSRC
Library . . . . . : KPSLIB
Last Change . . . . . : 03/01/89 14:10:21
Description . . . . . : *SAME
5738RG1 V2R1M0 910524          IBM SAA RPG/400          KPSLIB/RPGDRIVER  12/21/90 07:32:18  Page    2
SEQUENCE          IND  DO  LAST  PAGE  PROGRAM
NUMBER  *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE  NUM  UPDATE  LINE  ID
                S o u r c e   L i s t i n g
100  H          11/03/87
200  *  SAMPLE PROGRAM TO READ A RECORD, THEN CALL PROGRAM          02/23/89
300  *  'QY2FTML' TO TRANSFER MEMBER, REPEAT UNTIL LAST          02/23/89
400  *  RECORD, THEN PRINT LISTING.          02/23/89
500  FFTTEST IP F 120 DISK          11/10/87
600  FQSYSVRT O F 0132 OF PRINTER          02/23/89
700  I*          11/03/87
800  IFTTEST NS          11/11/87
* 4137          4137-**
900  I          1 3 INDEX          11/12/87
1000 I          4 4 OPTION          02/23/89
1100 I          5 5 REPL          11/12/87
1200 I          10 19 FRMLIB          02/23/89
1300 I          20 29 FRMFIL          02/23/89
1400 I          30 39 FRMMBR          02/23/89
1500 I          40 45 TYPE          02/23/89
1600 I          50 59 TOLIB          02/23/89
1700 I          60 69 TOFIL          02/23/89
1800 I          70 79 TOMBR          02/23/89
1900 I          80 85 TONATE          02/23/89
2000 I          90 97 RMTLOC          02/23/89
2100 I          98 107 PASSWD          02/23/89
2200 C*          11/03/87
2300 C          QYLIST  PLIST          11/11/87
2400 C          PARM          OPTION 1          02/23/89
2500 C          PARM          FRMLIB 10          02/23/89
2600 C          PARM          FRMFIL 10          02/23/89
2700 C          PARM          FRMMBR 10          02/23/89
2800 C          PARM          TYPE 6          02/23/89
2900 C          PARM          TOLIB 10          02/23/89

```

Figure E-8 (Part 1 of 3). RPG/400 Coding for File Transfer Support

```

3000 C          PARM          TOFIL 10          02/23/89
3100 C          PARM          TOMBR 10          02/23/89
3200 C          PARM          TODATE 6          02/23/89
3300 C          PARM          REPL 1           11/03/87
3400 C          PARM          RMTLOC 8          02/23/89
3500 C          PARM          PASSWD 10         02/23/89
3600 C          PARM          RCODE 1           02/23/89
3700 C          PARM          MSGNUM 8          02/23/89
3800 C*
3900 C          CALL 'QY2FTML' QYLIST          11/11/87
4000 C*
4100 QSYSVRT H 1 1P          11/03/87
                                02/23/89

```

```

5738RG1 V2R1M0 910524          IBM SAA RPG/400          KPSSLIB/RPGDRIVER          12/21/90 07:32:18          Page 3
SEQUENCE          IND          DO          LAST          PAGE          PROGRAM
NUMBER          *...1...+...2...+...3...+...4...+...5...+...6...+...7...* USE          NUM          UPDATE          LINE          ID
4200 0          OR          OF          02/23/89
4300 0          11 'FRMLIB'          02/23/89
4400 0          22 'FRMFIL'          02/23/89
4500 0          33 'FRMMBR'          02/23/89
4600 0          42 'TYPE'          02/23/89
4700 0          52 'TOLIB'          02/23/89
4800 0          63 'TOFIL'          02/23/89
4900 0          74 'TOMBR'          02/23/89
5000 0          86 'TODATE'          02/23/89
5100 0          95 'OPTION'          02/23/89
5200 0          101 'REPL '          02/23/89
5300 0          107 'RMTLOC'          02/23/89
5400 0          116 'RCODE '          02/23/89
5500 0          122 'MSGNUM'          02/23/89
5600 0          H 1 1P          02/23/89
5700 0          OR          OF          02/23/89
5800 0          15 '-----'          11/12/87
5900 0          26 '-----'          11/12/87
6000 0          37 '-----'          11/12/87
6100 0          46 '-----'          11/12/87
6200 0          57 '-----'          11/12/87
6300 0          68 '-----'          11/12/87
6400 0          79 '-----'          11/12/87
6500 0          88 '-----'          11/12/87
6600 0          95 '-----'          02/23/89
6700 0          100 '-----'          02/23/89
6800 0          109 '-----'          02/23/89
6900 0          115 '-----'          02/23/89
7000 0          124 '-----'          02/23/89
7100 0          D 2 N1P          02/23/89
7200 0          INDEX 3          02/23/89
7300 0          FRMLIB 15          02/23/89
7400 0          FRMFIL 26          02/23/89
7500 0          FRMMBR 37          02/23/89
7600 0          TYPE 44          02/23/89
7700 0          TOLIB 57          02/23/89
7800 0          TOFIL 68          02/23/89
7900 0          TOMBR 79          02/23/89
8000 0          TODATE 86          02/23/89
8100 0          OPTION 92          02/23/89
8200 0          REPL 98          02/23/89
8300 0          RMTLOC 109          02/23/89
8400 0          RCODE 114          02/23/89
8500 0          MSGNUM 124          02/23/89
***** END OF SOURCE *****
Additional Diagnostic Messages
* 7086          500 RPG PROVIDES BLOCK OR UNBLOCK SUPPORT FOR FILE FTTEST.
5738RG1 V2R1M0 910524          IBM SAA RPG/400          KPSSLIB/RPGDRIVER          12/21/90 07:32:18          Page 4
Cross Reference

```

File and Record References:

FILE/RCD	DEV/RCD	REFERENCES (D=DEFINED)
01 FTTEST	DISK	500D 800
02 QSYSVRT	PRINTER	600D 4100 5600 7100

Field References:

FIELD	ATTR	REFERENCES (M=MODIFIED D=DEFINED)
FRMFIL	A(10)	1300D 2600D 7400
FRMLIB	A(10)	1200D 2500D 7300
FRMMBR	A(10)	1400D 2700D 7500
INDEX	A(3)	900D 7200
MSGNUM	A(8)	3700D 8500
OPTION	A(1)	1000D 2400D 8100
PASSWD	A(10)	2100D 3500D

Figure E-8 (Part 2 of 3). RPG/400 Coding for File Transfer Support

```

QYLIST    PLIST  23000 3900M
RCODE     A(1)   36000 8400
REPL      A(1)   11000 33000 8200
RMTLOC    A(8)   20000 34000 8300
TODATE    A(6)   19000 32000 8000
TOFIL     A(10)  17000 30000 7800
TOLIB     A(10)  16000 29000 7700
TOMBR     A(10)  18000 31000 7900
TYPE      A(6)   15000 28000 7600
'QY2FTML' LITERAL 3900

```

Indicator References:

```

INDICATOR REFERENCES (M=MODIFIED D=DEFINED)
LR          5000
OF          6000 4200 5700
IP          4100 5600 7100

```

```

***** END OF CROSS REFERENCE *****
5738RG1 V2R1M0 910524 IBM SAA RPG/400 KPSLIB/RPGDRIVER 12/21/90 07:32:18 Page 5

```

Message Summary

```

* QR64137 Severity: 00 Number: 1
  Message . . . . : Record-Identifying-Indicator entry is blank.
* QR67086 Severity: 00 Number: 1
  Message . . . . : RPG handles blocking function for file. INFDS
                    updated only when blocks of data transferred.

```

```

***** END OF MESSAGE SUMMARY *****
5738RG1 V2R1M0 910524 IBM SAA RPG/400 KPSLIB/RPGDRIVER 12/21/90 07:32:18 Page 6

```

Final Summary

```

Message Count: (by Severity Number)
TOTAL      00      10      20      30      40      50
           2       2       0       0       0       0

```

Program Source Totals:

```

Records . . . . . : 85
Specifications . . . . . : 78
Table Records . . . . . : 0
Comments . . . . . : 7

```

PRM has been called.

Program RPGDRIVER is placed in library KPSLIB. 00 highest Error-Severity-Code.

```

***** END OF COMPILATION *****

```

Figure E-8 (Part 3 of 3). RPG/400 Coding for File Transfer Support

Calling File Transfer Support for a CL Program

Figure E-9 is an example of a CL program that uses file transfer support to retrieve a database member from a remote system and store it in the library QTEMP. The program uses the

Display Physical File Member (DSPPFM) command which allows the user to view the member. The user then has the option to submit the database member as a batch job or end the program.

```

5738SS1 V2R1M0 910524          Control Language          KPSLIB/SBMRMTJOB      12/21/90 08:42:43      Page 1

Program . . . . . : SBMRMTJOB
Library . . . . . : KPSLIB
Source file . . . . : QCLSRC
Library . . . . . : KPSLIB
Source member name . . . . . : SBMRMTJOB 03/21/89 10:15:10
Source printing options . . . . . : *SOURCE *XREF *GEN *NOSECLVL
Program generation options . . . . . : *NOLIST *NOXREF *NOPATCH
User profile . . . . . : *USER
Program logging . . . . . : *JOB
Allow RTVCLSRC command . . . . . : *YES
Replace program . . . . . : *YES
Target release . . . . . : *CURRENT
Authority . . . . . : *LIBCRTAUT
Text . . . . . : Retrieves a member from a remote sys and SBMDBJOB
Compiler . . . . . : IBM AS/400 Control Language Compiler

Control Language Source
SEQNBR *...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+. DATE
100-
200- /*****/
300- /* This program uses File Transfer Support to retrieve a data base */ 01/24/89
400- /* member from a remote system and store it in library QTEMP. The */
500- /* Display Physical File Member(DSPPFM) command is then used to */
600- /* allow the user to view the member. When the user is finished */
700- /* viewing the member "enter" is pressed to continue. An inquiry */
800- /* message is then sent to the user which allows their choice */
900- /* whether or not to submit this member as a batch job. If a "Y" */ 01/24/89
1000- /* response is given the Submit Data Base Job(SBMDBJOB) command is */
1100- /* used to submit the job to batch. If a "N" response is given */
1200- /* the program ends. */
1300- /*****/
1400-
1500- PGM PARM(&FROMLIB.&FROMFILE &FROMMBR)
1600- DCL VAR(&FROMLIB) TYPE(*CHAR) LEN(10)
1700- DCL VAR(&FROMFILE) TYPE(*CHAR) LEN(10)
1800- DCL VAR(&FROMMBR) TYPE(*CHAR) LEN(10)
1900- DCL VAR(&PASSWORD) TYPE(*CHAR) LEN(10) 02/24/89
2000- DCL VAR(&RTNCODE) TYPE(*CHAR) LEN(1) VALUE(' ')
2100- DCL VAR(&SBMJOB) TYPE(*CHAR) LEN(1) 01/24/89
2200-
2300- /*****/
2400- /* Retrieve user's password from a data area. */ 01/24/89
2500- /* (used to preserve security). */
2600- /*****/
2700- RTVDTAARA DTAARA(PASSWORD (1 10)) RTNVAR(&PASSWORD)
2800- 01/24/89
2900- /*****/
3000- /* Retrieve the member from the remote system using File */ 01/24/89
3100- /* Transfer Support. Note the call to File Transfer is */
3200- /* made with both CL variables and constant values. Also */
3300- /* note that File Transfer parameters are all positional. */
3400- /* You must, therefore, reserve space in the call for all */

5738SS1 V2R1M0 910524          Control Language          KPSLIB/SBMRMTJOB      12/21/90 08:42:43      Page 2

Control Language Source
SEQNBR *...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8 ...+... 9 ...+. DATE
3500- /* 14 parameters. If a parameter is not used, fill its */
3600- /* space with blanks. */
3700- /*****/
3800- CALL PGM(QSYS/QY2FTML) PARM(R &FROMLIB + 03/21/89
3900- &FROMFILE &FROMMBR ' ' QTEMP ' ' - 03/21/89
4000- ' ' ' Y RCHAS365 &PASSWORD &RTNCODE ' ') 03/21/89
4100- /*****/
4200- /* Check the FTS return code to insure a good completion. */
4300- /*****/
4400- IF COND(&RTNCODE *EQ '0') THEN(DO)
4500-

```

Figure E-9 (Part 1 of 2). CL Coding for File Transfer Support

```

4600-  /*****/
4700-  /* Display the member. */
4800-  /*****/
4900-      DSPPFM      FILE(QTEMP/&FROMFILE) MBR(&FROMMBR)
5000-
5100-  /*****/
5200-  /* Send the inquiry message to the user. */
5300-  /*****/
5400-      SHDUSRMSG  MSG('Submit the job to batch (Y,N)') VALUES(Y +
5500-                  N) DFT(Y) MSGRPY(&SBMJOB)
5600-
5700-  /*****/
5800-  /* Check the user's response. If it is "Y", submit the job */
5900-  /* to batch using the Submit Data Base Job(SBMDBJOB) command.*/
6000-  /* If the response is "N", return to caller. */
6100-  /*****/
6200-      IF          COND(&SBMJOB *EQ 'Y') THEN(SBMDBJOB +
6300-                  FILE(QTEMP/&FROMFILE) MBR(&FROMMBR))
6400-
6500-      ENDDO      /* FTS return code is good */
6600-      ENDPGM

```

```

01/24/89
01/24/89
01/24/89
03/21/89
01/24/89
01/24/89
01/24/89
01/24/89
01/24/89
03/21/89

```

***** END OF SOURCE *****

5738SS1 V2R1M0 910524 Control Language KPSLIB/SBMRMTJOB 12/21/90 08:42:43 Page 3

Cross Reference

Declared Variables

Name	Defined	Type	Length	References
&FROMFILE	1700	*CHAR	10	1500 3800 4900 6200
&FROMLIB	1600	*CHAR	10	1500 3800
&FROMMBR	1800	*CHAR	10	1500 3800 4900 6200
&PASSWORD	1900	*CHAR	10	2700 3800
&RTNCODE	2000	*CHAR	1	3800 4400
&SBMJOB	2100	*CHAR	1	5400 6200

* CPD0791 00 No labels used in program.

***** END OF CROSS REFERENCE *****

* CPD0772 00 Program contains commands only valid when run interactively.

5738SS1 V2R1M0 910524 Control Language KPSLIB/SBMRMTJOB 12/21/90 08:42:43 Page 4

Message Summary

Total	0-9	10-19	20-29	30-39	40-49	50-59	60-69	70-79	80-89	90-99
2	2	0	0	0	0	0	0	0	0	0

Program SBMRMTJOB created in library KPSLIB. Maximum error severity 00.

***** END OF MESSAGE SUMMARY *****
***** END OF COMPILATION *****

Figure E-9 (Part 2 of 2). CL Coding for File Transfer Support

File Transfer Support Messages

File Transfer Support messages give more information about the error condition that occurred on either the local or remote system. The FTS message number itself is available to the application via the Message-ID parameter. The FTS message is logged in the program message queue of the program using FTS and contains a description of the error that occurred.

If a local system error occurs, the FTS message directly identifies the error that occurred.

If a remote system error occurs, FTS-1007 will be logged in the program message queue. The actual remote system error is identified in the message text associated with FTS-1007, and returned to the program in the Message-ID parameter. If the remote system is a AS/400 system, you need to examine the message text of the system message associated with the

remote FTS message. The correlation between the system message ID and the FTS message ID are described below. If the remote system is a System/36, the remote FTS message ID is described in the *Using S/36 Communications* manual.

Figure E-10 (Page 1 of 2). File Transfer Messages

FTS Message ID	System Message ID
FTS-1001	CPI7A01
FTS-1002	CPI7A02
FTS-1003	CPD7A03
FTS-1004	CPD7A04
FTS-1005	CPD7A05
FTS-1006	CPD7A06
FTS-1007	CPD7A07
FTS-1008	CPD7A08
FTS-1009	CPD7A09

Figure E-10 (Page 2 of 2). File Transfer Messages

FTS Message ID	System Message ID
FTS-1010	CPI7A10
FTS-1011	CPI7A11
FTS-1012	CPD7A12
FTS-1013	CPD7A13
FTS-1014	CPF7A14
FTS-1019	CPD7A19
FTS-1020	CPD7A20
FTS-1021	CPD7A21
FTS-1022	CPD7A22
FTS-1023	CPD7A23
FTS-1024	CPD7A24
FTS-1025	CPD7A25
FTS-1026	CPD7A26
FTS-1027	CPD7A27
FTS-1028	CPD7A28
FTS-1030	CPD7A30
FTS-1031	CPD7A31
FTS-1032	CPD7A32
FTS-1033	CPD7A33
FTS-1034	CPD7A34
FTS-1035	CPD7A35
FTS-1036	CPD7A36
FTS-1037	CPD7A37
FTS-1038	CPD7A38
FTS-1039	CPD7A39
FTS-1040	CPD7A40
FTS-1041	CPD7A41
FTS-1042	CPD7A42
FTS-1043	CPD7A43
FTS-1044	CPD7A44
FTS-1046	CPD7A46
FTS-1049	CPD7A49
FTS-1050	CPD7A50
FTS-1051	CPD7A51
FTS-1052	CPD7A52
FTS-1053	CPD7A53
FTS-1054	CPD7A54
FTS-1055	CPD7A55
FTS-1056	CPD7A56
FTS-1057	CPD7A57

Figure E-10 (Page 2 of 2). File Transfer Messages

FTS Message ID	System Message ID
FTS-1058	CPD7A58
FTS-1059	CPD7A59
FTS-1060	CPD7A60
FTS-1063	CPD7A63
FTS-1064	CPD7A64
FTS-1065	CPD7A65
FTS-1066	CPD7A66
FTS-1067	CPD7A67
FTS-1068	CPD7A68
FTS-1069	CPD7A69
FTS-1070	CPD7A70
FTS-1071	CPD7A71

Bibliography

The following manuals contain information you may need. The manuals are listed with their full title and order number. When these manuals are referred to in this guide, the short title listed is used. Except where otherwise indicated, each is an AS/400 system manual.

Communications Manuals

- *Communications: Advanced Peer-to-Peer Networking Guide*, SC41-8188, provides the information necessary to define and use the AS/400 system advanced peer-to-peer networking (APPN) function. This manual contains information on both ICF and Common Programming Interface (CPI) Communications. **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 manual is a guide for developing application programs and for defining the communications environment for APPC communications. This manual contains information on both ICF and Common Programming Interface (CPI) Communications. **Short Title:** *APPC Programmer's Guide*.
- *Communications and Systems Management Guide (Alerts and Distributed Systems Node Executive)*, SC41-9661, contains information about preparing a system for remote management activities and about using the change management feature distributed systems node executive (DSNX). **Short Title:** *Alerts and DSNX Guide*.
- *Communications: Asynchronous Communications Programmer's Guide*, SC41-9592, provides the application programmer with information for creating an asynchronous communications definition, writing programs that use asynchronous communications, and responding to return codes. It also provides information on developing asynchronous communications application programs that use ICF. **Short Title:** *Asynchronous Communications Programmer's Guide*.
- *Communications: BSC Equivalence Link Programmer's Guide*, SC41-9593, provides the application programmer or programmer with the information needed to write programs that use binary synchronous communications equivalence link (BSCEL) to communicate with a remote system. It also contains information for programmers about other systems and devices that communicate with BSCEL on the AS/400 system. The manual describes how to set up BSCEL and how to run application programs that use BSCEL. **Short Title:** *BSC Equivalence Link Programmer's Guide*.
- *Communications: Finance Communications Programmer's Guide*, SC41-8099, provides information for the application programmer or system administrator who uses the OS/400 finance support for financial communications between devices at multiple locations. It describes how finance support communicates with a controller and how to set up finance support. This manual provides information for writing application programs to communicate with application programs on the finance controller using ICF. **Short Title:** *Finance Communications Programmer's Guide*.
- *Communications: Intrasystem Communications Programmer's Guide*, SC41-9864, provides the application programmer or programmer with information about interactive communications between two application programs on the same AS/400 system. This manual also provides information on developing intrasystem communications application programs that use ICF. **Short Title:** *Intrasystem Communications Programmer's Guide*.
- *Communications: Management Guide*, SC41-0024, provides the system operator or programmer with information on using AS/400 communications such as work management, communications status, error handling, aggregate line speed and subsystem storage. **Short Title:** *Communications Management Guide*.
- *Communications: Operating System/400* Communications Configuration Reference*, SC41-0001, contains information for configuring objects for communications. **Short Title:** *OS/400* Communications Configuration Reference*.
- *Communications: Remote Job Entry User's Guide and Reference*, SC09-1168, provides information for using the Communications Utilities remote job entry (RJE) to submit jobs to an IBM host processor. **Short Title:** *RJE User's Guide and Reference*.
- *Communications: Retail Communications Programmer's Guide*, SC41-9858, provides information for setting up and starting retail communications between devices at multiple locations. **Short Title:** *Retail Communications Programmer's Guide*.
- *Communications: SNA Upline Facility Programmer's Guide*, SC41-9594, contains information on using the Systems Network Architecture Upline Facility with the IBM AS/400 system. **Short Title:** *SNA Upline Facility Programmer's Guide*.

- *Communications: 3270 Device Emulation Guide*, SC41-9602, provides information on setting up and starting either 3270 Binary Synchronous Communication (BSC) or Systems Network Architecture (SNA) device emulation. **Short Title:** *3270 Device Emulation Guide*.
- *Guide to Programming Application and Help Displays*, SC41-0011, contains information on how to program for application and help text displays. **Short Title:** *Guide to Programming Displays*.
- *Network Planning Guide*, GC41-9861, provides an overview of network planning tasks and helps you determine which communications functions to use. This manual is a starting point for communications planning. **Short Title:** *Network Planning Guide*.
- *System Programmer's Communications Interface Guide*, SC41-0027, provides user-defined communications information for the AS/400 system. **Short Title:** *System Programmer's Communications Interface Guide*.
- *Transmission Control Protocol/Internet Protocol Guide*, SC41-9875, provides information on TCP/IP configurations, IBM-supplied applications, and user-written applications. **Short Title:** *TCP/IP Guide*.

Programming Language Manuals

- *Data Description Specifications Reference*, SC41-9620, contains information about coding data description specifications for physical, logical, display, printer, and ICF files. **Short Title:** *DDS Reference*.
- *Data Management Guide*, SC41-9658, provides information to help the system programmer manage key aspects of the system. For example, it describes how to use diskette and tape files. **Short Title:** *Data Management Guide*.
- *Database Guide*, SC41-9659, provides information about the AS/400 database management system, and describes how to set up and use a database on the AS/400 system. **Short Title:** *Database Guide*.
- *Distributed Data Management Guide*, SC41-9600, contains the information needed to use DDM on a network. It includes AS/400 system DDM concepts, preparing for DDM communications, and all DDM-related programming information needed by the DDM programmer. **Short Title:** *DDM Guide*.
- *Guide to Programming for Printing*, SC41-8194, provides information to help the programmer manage key aspects of the system. For example, it describes how to use printer files. **Short Title:** *Guide to Programming for Printing*.
- *Languages: Systems Application Architecture* AD/Cycle* COBOL/400* Reference*, SC09-1380,

provides a description of the COBOL/400 language organization, program organization, procedure division statements, and compiler directing statements. **Short Title:** *COBOL/400* Reference*.

- *Languages: Systems Application Architecture* AD/Cycle* COBOL/400* User's Guide*, SC09-1383, provides the information needed to write, test and maintain COBOL/400 programs for the AS/400 system. **Short Title:** *COBOL/400* User's Guide*.
- *Languages: Systems Application Architecture* AD/Cycle* RPG/400* User's Guide*, SC09-1348, provides the information needed to use the RPG/400 programming language to code programs for the AS/400 system. **Short Title:** *RPG/400* User's Guide*.
- *Languages: Systems Application Architecture* C/400* User's Guide*, SC09-1347, provides the information needed to use the C/400 programming language to code programs for the AS/400 system. **Short Title:** *C/400* User's Guide*.
- *Languages: Systems Application Architecture* FORTRAN/400* Reference*, SC41-9844, provides information on using the FORTRAN/400 programming language. **Short Title:** *FORTRAN/400* Reference*.
- *Languages: Systems Application Architecture* FORTRAN/400* User's Guide*, SC41-9845, provides information about entering and coding a FORTRAN/400 program, compiling a source program, performing input/output operations, running and debugging the program, and using interprogram communication in the FORTRAN environment. **Short Title:** *FORTRAN/400* User's Guide*.
- *Programming: Control Language Programmer's Guide*, SC41-8077, provides a wide-ranging discussion of AS/400 system programming topics. **Short Title:** *CL Programmer's Guide*.
- *Programming: Control Language Reference*, SC41-0030, provides information on control language commands. **Short Title:** *CL Reference*.
- *Programming: Extended Program Model User's Guide and Reference*, SC09-1294, contains information about open feedback and I/O feedback areas for the C/400 language. **Short Title:** *Extended Program Model User's Guide and Reference*.
- *Programming: Work Management Guide*, SC41-8078, contains information on how to create and change a work management environment. **Short Title:** *Work Management Guide*.
- *Security Concepts and Planning*, SC41-8083, contains information about general security concepts for the system. **Short Title:** *Security Concepts and Planning*.
- *System Operator's Guide*, SC41-8082, provides information for the system operator on how to use

the system unit control panel. **Short Title:**
Operator's Guide.

System/36-Related Manuals

- *System/36 System Reference*, SC21-9020 contains information on System/36 naming conventions you might need for file transfer support (FTS).
- *Using S/36 Communications*, contains information on the correlation between system message IDs and File Transfer Support message IDs on the System/36.

Index

A

acquire operations

- acquire 5-2
- automatic 5-3
- description 3-6
- explicit 5-2
- implicit 5-2
- language operations A-2
- programming considerations 8-2
- reasons for failure 5-3
- source program device 5-3
- target program device 5-3

acquire program device (ACQPGMDEV) parameter

- description 4-8
- open operation 5-1
- use 4-4
- values 4-4

acquiring sessions

- description 3-5
- elements used 3-17
- sequence diagrams 3-17
- starting a session 3-17

Add Intersystem Communications Function Device

Entry (ADDICFDEVE) command

- description 4-3
- example 3-8, 6-22
- location name parameter 3-6
- use 4-9

Add Prestart Job Entry (ADDPJE) command 8-8

advanced peer-to-peer networking (APPN), definition 2-2

advanced program-to-program communications (APPC)

- definition 2-1
- introduction 2-1
- selecting record formats 5-9
- VRYPFG command 3-3, 3-14

allow write (ALWWRT) function

- example 6-15
- use 6-15

APPC

- See advanced program-to-program communications (APPC)

application programs

- definition 3-2
- description 3-2
- started by AS/400 system 3-1, 3-4
- started by remote system 3-1, 3-7

APPN

- See advanced peer-to-peer networking (APPN)

asynchronous communication

- definition 2-2
- ending 3-14
- introduction 2-2

asynchronous communication (*continued*)

- starting 3-5
- VRYPFG command 3-14

AS/400 system

- configuration 2-4
- control language (CL) 2-5
- data description specifications (DDS) 2-5
- languages supported 2-6
- OS/400 program 2-4
- security 2-5

attributes

- communications file 4-4, 4-14
- device dependent 4-14
- ICF file 4-4, 4-14

B

batch data transfer example 10-2

batch transmission example 6-4

BSC equivalence link (BSCSEL)

- definition 2-2
- introduction 2-2
- VRYPFG command 3-3, 3-14

BSC MSRJ (non-ICF communications) 2-4

BSCSEL

- See BSC equivalence link (BSCSEL)

C

cancel an invite (CNLINVITE) keyword

- example 6-16
- format 6-16
- use 6-16

cancel function

- canceling sent records 7-10
- COBOL/400 program 7-10
- COBOL/400 WRITE statement example 7-11
- C/400 write statement example 7-11
- description 6-10
- example 6-10
- RPG/400 output specification 7-11

cancel invite (\$CNLINV) system-supplied format

- COBOL/400 WRITE statement example 7-17
- C/400 write statement example 7-17
- description 7-17
- example 7-17
- RPG/400 example 7-17

- sending error signal example 7-10

cancel with invite (\$CANL) system-supplied format 7-6

cancel (\$CANLNI) system-supplied format 7-10

canceling

- invite of program device 5-17
- sent records 7-10

casual inquiry menu (CIMENU) 10-35, 11-36

Change Intersystem Communications Function Device Entry (CHGICFDEVE) command
 description 4-2
 use 4-9

Change Intersystem Communications Function File (CHGICFF) command
 description 4-2
 use 4-7

Change Prestart Job Entry (CHGPJE) command 8-8

change transmission direction request 6-19

charts
 DDS keyword processing 6-24
 summary 6-22

checking return codes 5-7

CIMENU
 See casual inquiry menu (CIMENU)

CL
 See control language (CL)

close considerations 8-4, 8-5

close operation 5-19

closing an ICF file 5-19

COBOL/400 programming language
 DDS for display file (DSPFIL) 10-34
 display file 10-25, 10-32
 ending
 sessions 7-20
 transactions 7-18
 error handling 10-2
 fail function 7-8, 7-9
 feedback areas 10-2
 file transfer example E-21
 indicating error conditions 7-8
 interface 10-1
 introduction 10-1
 output operations 7-4
 request to write 7-16
 request-to-write 7-14
 sending
 data 7-4
 negative-responses 7-12

commands
 ADDICFDEVE 4-3
 CHGICFDEVE 4-3, 4-9
 CHGICFF 4-2, 4-7
 CRTICFF 4-2
 CRTSBSD 8-5
 DLTF 4-2
 DLTOVR 4-2
 DLTOVRDEVE 4-3
 DSPFD 4-3
 DSPFFD 4-3
 DSPOVR 4-3, 4-8
 ENDMOD 3-14
 file-level attribute type 4-2
 information display type 4-3
 OVRICFDEVE 4-2, 4-3
 OVRICFF 4-2, 4-7

commands (continued)
 program device entry type 4-3
 RMVICFDEVE 4-3, 4-9
 STRMOD 3-3
 summary 4-17
 types 4-2
 VRYCFG 3-3, 3-14

communication type, definition 2-1

communications
 application considerations using ICF file 8-1
 asynchronous 2-2
 configuration 2-4, 3-2
 displaying communications status 12-7
 ending 3-12, 6-16
 entries
 changing 8-5
 default user specification 8-5
 description 8-5
 features 2-1
 lines
 AS/400 system support 2-4
 introduction 2-4
 link
 connecting 3-5
 establishing 3-5
 example 3-7
 operating the system 1-1
 operations
 codes A-2
 purpose 2-1
 system 1-1
 processing using DDS keywords 6-1
 program-to-program 3-1
 sessions 3-13
 starting 3-5
 transactions 3-8
 types
 combinations 2-4
 description 3-2
 introduction 2-1

communications configurations 2-4, 3-2

communications session, definition 3-5

communications transaction, definition 3-7

communications type (CMNTYPE) parameter
 use 4-16
 values 4-16

compare value, RECID keyword 5-9

configuration
 data communications 1-1
 description 2-4, 3-2
 example relationship 3-4
 types 3-2
 varying on 3-3
 VRYCFG description 3-3

configuring for communications 3-2

confirm request received 6-18

confirm (CONFIRM) keyword 6-3

considerations

- close 8-4, 8-5
- end-of-session 8-4
- file 8-13
- input 8-3
- language operations A-2
- output 8-2
- programming
 - description 8-1
 - file overrides 8-13
 - file redirection 8-13
 - handling program start requests 8-5
 - minor 8-1
 - remote environment definition 8-5
 - remote program start request 8-5
 - return codes 8-1
 - security 8-12
 - subsystem creation 8-5
 - system 8-12
- release 8-4

control language (CL)

- acquire example 3-6
- introduction 2-5

Create Class (CRTCLS) command

- description 8-12
- ICF file specification 8-12
- use 8-12

Create Intersystem Communications Function File (CRTICFF) command

- description 4-2
- parameter values 4-7

Create Subsystem Description (CRTSBSD) command 8-5

C/400 programming language

- accessing feedback areas 9-2
- DDS for display file (DSPFIL) 9-10
- display file 9-10
- error handling 9-2
- file transfer example E-14
- function 9-1
- interface 9-1
- introduction 9-1
- multiple session 9-2

D

data

- management
 - calculating I/O buffers 4-5
 - example 3-2
 - ICF 3-2
 - maximum record length (MAXRCLEN) 4-5
- parameter 4-5
- receiving
 - description 3-11
 - example 6-7, 7-6
 - how to 6-7, 7-6
 - in a file E-1
- sending
 - batch transmission example 6-4

data (continued)

sending (continued)

- by output operation, example 3-11
- confirm (CONFIRM) 6-4
- description 3-11, 5-6
- example 3-1, 6-6
- force-data (FRCDTA) 6-4
- format-name (FMTNAME) 6-5
- function-management-header (FMH) 6-5
- in a file E-1
- subdevice-selection (SUBDEV) 6-5
- using DDS keywords 6-4
- variable length data (VARLEN) 6-4

transfer E-1

data description specifications (DDS)

- description 2-5
- example file 6-19
- functions 4-2
- keywords
 - allow-write (ALWWRT) 6-15
 - cancel (CANCEL) 6-10
 - cancel-invite (CNLINVITE) 6-16
 - confirm (CONFIRM) 6-4
 - detach (DETACH) 6-16
 - end-of-group (ENDGRP) 6-5
 - end-of-session (EOS) 6-17
 - evoke (EVOKE) 6-1
 - fail (FAIL) 6-9
 - force-data (FRCDTA) 6-4
 - format-name (FMTNAME) 6-5
 - function-management-header (FMH) 6-5
 - introduction 6-1
 - invite (INVITE) 6-7, 7-6
 - map to system-supplied formats 7-23
 - negative-response (NEGRSP) 6-11
 - processing 6-1
 - processing charts 6-22
 - receive confirm (RCVCONFIRM) 6-18
 - receive detach (RCVDETACH) 6-19
 - receive end of group (RCVENDGRP) 6-19
 - receive fail (RCVFAIL) 6-19
 - receive negative response (RCVNENGRSP) 6-19
 - receive turnaround (RCVTRNRND) 6-19
 - receive-cancel (RCVCANCEL) 6-19
 - receive-function-management-header (RCVFMH) 6-19
 - record-identification (RECID) 6-8
 - record-identifier selection (RECID) 5-9
 - request-to-write (RQSWRT) 6-13
 - respond-to-confirm (RSPCONFIRM) 6-12
 - security (SECURITY) 6-2
 - subdevice selection (SUBDEV) 6-5
 - synchronization level (SYNLVL) 6-3
 - timer (TIMER) 6-8
 - used to send data 6-4
 - variable length data record (VARLEN) 6-4
- response keywords 6-18
- source 9-23

DDS

See data description specifications (DDS)

default user (DFTUSR) parameter 8-5

Delete Override (DLTOVR) command 4-2, 4-8

detach function

example 3-12

system-supplied formats 7-18

use 7-18

detach (DETACH) keyword 6-17

detail inquiry menu (DTLMNU)

description 11-36

use 10-35

determining the wait-for-record value 4-6

device file 4-1

device (DEV) parameter 4-12

Display Active Prestart Jobs (DSPACTPJ)

command 8-8

display file

DDS COBOL/400 program

description 10-32

example 10-34

DDS C/400 program

description 9-10

example 9-10

DDS RPG/400 program

description 11-34

example 11-35

display information commands 4-3

Display Override (DSPOVR) command

description 4-3

use 4-3, 4-8

displaying communications status 12-7

DTLMNU

See detail inquiry menu (DTLMNU)

E

End Mode (ENDMOD) command 3-14

End Prestart Job (ENDPJE) command 8-8

end-of-group (ENDGRP) keyword 6-5

end-of-session function

considerations 8-4

description 6-17

examples

C/400 write statement 7-19, 7-21

release operation 6-17

system-supplied formats 7-20

use 7-20

end-of-session (\$\$EOS) system-supplied format

COBOL/400 WRITE statement example 7-21

C/400 write statement example 7-21

description 7-20

example 7-20

RPG/400 example 7-21

end-of-transaction operation 7-18

ending

communications

with a remote system 3-12

with a target system 3-12

ending (continued)

sessions

communications 6-17

description 3-13

end-of-session function 7-20

release operation 6-17, 7-20

transactions 3-12, 6-16

error

handling 2-5

processing 9-2

recovery 2-5

establishing a session with a requesting program

device 3-8

Ethernet 2-3, 2-4

evoke

parameter list

format 7-3

use 7-3

parameters 3-7

evoke function

COBOL/400 write statement example 7-3

C/400 write statement example 7-3

description 7-2

evoke with detach (\$\$EVOKET) system-supplied

format 7-2

evoke with invite (\$\$EVOK) system-supplied

format 7-2

evoke (\$\$EVOKNI) system-supplied format 7-2

example 6-1

parameters 6-1

RPG/400 send example 7-3

security information 6-1

starting remote programs 7-2

evoke with detach (\$\$EVOKET) system-supplied

format

description 7-2, 7-18

ending communications 7-18

evoke with invite (\$\$EVOK) system-supplied

format 7-2

evoke (EVOKE) keyword

format 6-1, 6-2

purpose 6-1

security information 6-2

use 6-1

evoke (\$\$EVOKNI) system-supplied format 7-2

examples

COBOL programming language

AS/400 system 10-2

batch data transfer description 10-4, 11-4

local program 10-2

file member qualifiers E-9, E-11

multiple session description 10-25, 11-26

multiple sessions overview 10-2, 11-3

RPG/400 ICF file to the AS/400 system 11-3

RPG/400 programming language

AS/400 system 11-3

batch data transfer description 10-4, 11-4

local program 11-5

F

FAIL function

- COBOL statement example 7-9
- C/400 statement example 7-9
- RPG/400 statement example 7-9

fail (FAIL) keyword

- format 6-9
- receiving example 6-9
- sending example operation 6-10
- use 6-9

fail (\$\$FAIL) system-supplied format

- COBOL/400 write statement example 7-9
- C/400 write statement example 7-9
- description 7-8
- example 7-8
- RPG/400 output specification example 7-9

feedback areas C-1

file

- attributes 4-4, 4-14
- changing an ICF file 4-7
- device 4-1, 4-4
- file members
 - receiving E-1
 - sending E-1
 - transferring E-1
- ICF 2-5
- overrides 8-13
- redirection 8-13
- support attributes 4-4
- with OVRICFF 4-7
- writing an ICF file program 5-1

file level attribute commands 4-2

file level definition 4-9

file transfer support (FTS)

- CL program example E-28
- COBOL/400 example E-21
- C/400 example E-14
- definition E-1
- examples of file member qualifiers E-9, E-11
- messages E-29
- parameter list E-4
- qualifiers E-4
- RPG/400 example E-25
- subsystems using E-2

finance

- ICF-supported communications 2-3
- non-ICF communications 2-4

finance communications, definition 2-3

force data (FRCDTA) keyword 6-4

format name (FMTNAME) keyword 6-5

format selection processing

- remote (RMTFMT) 5-9
- summary 5-9

format selection processing (FMSTLT) parameter

- description 5-9
- determining record format 5-16
- use 4-16
- values 4-16

FTS

See file transfer support (FTS)

function management header (FMH) keyword 6-5

G

get operation

See read operation

get-attributes operation

- language operations A-2
- status information fields 5-4
- use 5-4

H

high-level languages

- example program 2-6
- languages supported 2-6

I

ICF (intersystem communications function)

See intersystem communications function (ICF)

input

- considerations 8-3
- operations 6-7, 7-6

installation, data communications 1-1

integrated services digital network (ISDN), definition 2-4

interactive communications writing programs 1-1

intersystem communications function (ICF)

- acquire, description 3-6
- attributes 4-14
- closing 5-19
- commands
 - ADDICFDEVE 4-3
 - CHGICFDEVE 4-3
 - CHGICFF 4-2
 - CRTICFF 4-2
 - DLTF 4-2
 - DLTOVR 4-2
 - DLTOVRDEVE 4-3
 - DSPFD 4-3
 - DSPFDD 4-3
 - DSPOVR 4-3
 - file-level attribute type 4-2
 - information display type 4-3
 - OVRICFDEVE 4-2, 4-3
 - OVRICFF 4-2
 - program device entry type 4-3
 - RMVICFDEVE 4-3
 - summary 4-17
 - types 4-2
- communications operations 3-1
- communications types
 - introduction 2-1
 - purpose 2-1
 - support 3-2
- creating a file 6-19

intersystem communications function (ICF) (continued)

- data management, definition 3-2
- defining program devices to 4-8
- definition 1-1, 3-1
- description 3-1, 4-1
- end-of-session 6-17
- evoke example 6-1
- example configuration 3-4
- file
 - attributes 4-4
 - changing 4-7
 - closing 5-19
 - commands 4-2
 - configuration relationship 3-4
 - creating 4-2, 6-19
 - defining record formats 4-3
 - definition 3-3
 - description 2-5, 4-1
 - device 4-1, 4-2
 - maximum record length 4-5
 - opening 4-4, 5-1
 - overriding 4-7
 - overview example 4-1
 - QICDMF 4-2
 - requesting device input output following an
 - acquire 8-2
 - support attributes 4-4
 - target program input output following an
 - acquire 8-2
 - using 5-1
- file-level attribute type 4-2
- general description 1-1
- maximum number of program devices 4-4
- mode name 5-7
- negative response error data 5-7
- opening 5-1
- operations and functions
 - acquire 5-2
 - allow-write (ALWWRT) 6-15
 - cancel 6-10, 7-10
 - cancel-invite 6-16, 7-17
 - confirm 6-4
 - detach 3-12, 6-4, 7-18
 - end-of-group 6-5
 - end-of-session 3-13
 - end-of-session function 7-20
 - evoke, general description 6-1, 7-2
 - fail 6-9, 7-8
 - force-data 6-4
 - format-name 6-5
 - function-management header 6-5
 - how to use 6-1
 - invite 6-7, 7-6
 - negative-response 6-11, 7-12
 - read 6-7, 7-6
 - read-from-invited-program-devices 6-7, 7-6
 - release 5-19
 - request-to-write 6-13, 7-14
 - respond-to-confirm 6-12

intersystem communications function (ICF) (continued)

- operations and functions (continued)
 - select record format 6-8
 - subdevice-selection 6-5
 - summary chart 6-22, 7-20
 - timer 6-8, 7-7
 - variable-length 6-4
 - write 6-4, 7-4
- other communication types 2-4
- program start requests 3-19
- record formats 4-3
- remote format name 5-7
- remotely started sessions 3-19
- return code 5-7
- safe indicator 5-7
- using 5-1
- write request indication 5-7
- writing a program 5-1

intrasystem communications, definition 2-3

invite function

- description 6-7, 7-6
- example 5-12

invite (INVITE) keyword 6-7

inviting a program device 5-8

ISDN

See integrated services digital network (ISDN)

item inquiry menu (ITMMNU) 10-35

item inquiry screen 2 (ITMSC2) 11-37

item inquiry screen 3 (ITMSC3) 11-37

I/O feedback area 5-6, C-3

J

job ended signal 5-15

job-level definition 4-9

K

keyword processing charts 6-22

keywords

- additional 6-12
- problem notification
 - cancel (CANCEL) 6-10
 - fail (FAIL) 6-9
 - negative-response (NEGRSP) 6-11
- receiving data
 - invite (INVITE) 6-7
 - record-identification (RECID) 6-8
 - timer (TIMER) 6-8

L

LAN

See local area network (LAN)

language operations supported by ICF A-1

local area network (LAN)

- Ethernet 2-3
- token-ring 2-3

local location name (LCLLOCNAME) parameter 4-12
local system 5-20
locally-started sessions 3-17

M

major return codes 8-1
maximum length
 See maximum record length (MAXRCDLEN) parameter
maximum program device (MAXPGMDEV)
 parameter 4-5
maximum record length (MAXRCDLEN)
 parameter 4-3, 4-5
maximum wait interval 5-13
messages, file transfer E-29
minor return codes 8-1
mode (MODE) parameter 4-12
multiple
 sessions 3-16
 transactions 3-14
multiple-session inquiry example
 description for COBOL/400 program 10-25
 description for C/400 program 9-2

N

negative-response function
 COBOL/400 program 7-14
 C/400 program 7-14
 RPG/400 example 7-14
negative-response (NEGRSP) keyword 6-11
negative-response (\$\$NRSPNI) system-supplied
 format
 description 7-12
 sending error condition example 7-12
negative-response-with-invite (\$\$NRSP) system-
 supplied format
 description 7-6, 7-12
 output buffer requirements 7-13
network
 Ethernet 2-3
 token-ring 2-3
non-intersystem communications
 combinations of communications types 2-4
 remote work stations 2-4
 running on the AS/400 system 2-3
number of seconds value 4-6

O

open feedback area 5-2
open operations
 language operations A-2
 programming considerations 8-2
operations and functions
 allow-write (ALWWRT) 6-15
 cancel 6-10
 detach
 description 3-12

operations and functions (continued)

detach (continued)
 example 3-12
 return code checking 3-12
end-of-session 3-13
evoke
 COBOL/400 write statement example 7-3
 C/400 write statement example 7-3
 parameters 3-7
 RPG/400 send example 7-3
 use with DDS keywords 6-1
get 7-6
ICF 3-1
invite 6-7, 7-6
read 6-7
read-from-invited-program-devices 6-7, 7-6
request-to-write (RQSWRT) 6-13
output considerations 8-2
output operations
 COBOL/400 send data example 7-5
 C/400 send data example 7-5
 description 6-4
 RPG/400 send data example 7-6
 sending data (COBOL/400 programming
 language) 7-4

P

parameter list
 evoke 7-3
 file transfer E-4
 format 7-3
parameters
 ACQPGMDEV 4-8
 CMNTYPE 4-16
 DEV 4-12
 evoke 3-7
 evoke function 6-1
 FMTSLT 4-16
 LCLLOCNAME 4-12
 MODE 4-12
 NONE 4-4
 PGMDEV 3-6, 4-9
 RMTFMT 5-10
 RMTLOCNAME 4-11
 RMTNETID 4-12
 SECURE 4-16
 sent to the started program 3-19
planning data communications 1-1
positive-response (\$\$POSRSP) system-supplied
 format 7-14, A-6
prestart job entry
 application considerations 8-9
 commands
 ADDPJE 8-8
 CHGPJ 8-8
 CHGPJE 8-8
 DSPACTPJ 8-8
 ENDPJ 8-8
 RMVPJE 8-8

- prestart job entry** *(continued)*
 - commands *(continued)*
 - RTVDTAARA 8-9
 - STRPJ 8-8
 - description 8-8
 - program initialization parameters 8-9
 - security considerations 8-9
- problem determination, error detection** 12-1
- problem notification**
 - formats 7-8
 - functions 6-9
- processing**
 - override
 - example 4-8
 - methods 4-8
 - release operation 5-19
 - using DDS keywords 6-1
- program device (PGMDEV) parameter** 3-6, 4-9
- program devices**
 - acquire example 4-7
 - acquiring
 - description 5-2
 - source program device 5-3
 - target program device 5-3
 - when file is open 4-4
 - canceling an invite 5-17
 - defining
 - system-supplied entries 7-1
 - to an ICF file 4-8
 - definition 4-1
 - definition list 5-4
 - determining
 - maximum number 4-5
 - which has data 5-16
 - entries 4-1
 - error from one 5-16
 - file level entry 4-9
 - get attributes 5-4
 - invited signal 5-15
 - inviting 5-8
 - job level entry 4-9
 - levels 4-9
 - mapping to communications configurations 4-10
 - obtaining device information 5-4
 - program device entry commands 4-3
 - program device name 4-4
 - read from
 - description 5-11
 - example 5-12
 - one device 5-16
 - time out 5-15
 - reading from invited 5-11
 - releasing 5-19
 - specifying 4-1
 - writing then reading from one 5-17
 - writing to 5-8
- program start requests**
 - description of failures (message CPF1269) B-21

- program start requests** *(continued)*
 - failed B-21
 - introduction 3-4
 - prestart jobs 8-8
 - reason codes B-21
 - starts a session 3-19
- program start request, definition** 3-4
- programming considerations**
 - acquire 8-2
 - applications using ICF file
 - COBOL/400 language 10-1
 - C/400 language 9-1
 - RPG/400 language 11-1
 - close 8-5
 - communications using ICF file 8-1
 - defining environment 8-5
 - description 8-1
 - end-of-session 8-4
 - files
 - overrides 8-13
 - redirection 8-13
 - handling program start requests 8-5
 - input 8-3
 - major codes 8-1
 - minor codes 8-1
 - open 8-2
 - output 8-2
 - prestarting jobs 8-8
 - release 8-4
 - remote environment definition 8-5
 - remote program start request 8-5
 - return codes 8-1
 - security 8-12
 - subsystem creation 8-5
 - system 8-12
- programming languages**
 - COBOL summary charts
 - communications operations A-2
 - operation codes A-2
 - RPG/400 summary charts
 - communications operations A-2
 - operation codes A-2
- programs**
 - application definition 3-2
 - application started by AS/400 system 3-4
 - at a remote system 3-7, 3-19
 - evoke functions 7-2
 - link with target program, example 3-9
 - parameters sent to started program 3-19
 - start requests 3-19
 - starting
 - description 3-4
 - on the remote system 6-1
 - remote programs 7-2
 - starts remote session, example 3-17
- put operation**
 - See write operations

Q

QBASE subsystem 8-5

QCMN subsystem 8-5

QICDMF file

characteristics 7-1

purpose 4-2

use 7-1

R

read operation

description 6-7

example 5-17

language operations A-2

read-from-invited-program-device

data available 5-15

example 5-12, 6-7, 7-6

language operations A-2

operations 6-7

responses 5-15

read-from-invited-program-devices operation 6-7

reason codes (for failed program start requests) B-21

receive-cancel indication (RCVCANCEL)
keyword 6-19

receive-confirm (RCVCONFIRM) keyword 6-18

receive-detach (RCVDETACH) keyword 6-19

receive-end-of-group indication (RCVENDGRP)
keyword 6-19

receive-fail indication (RCVFAIL) keyword 6-19

receive-function-management-header indication (RCVFMH) keyword 6-19

receive-negative-response indication (RCVNEGRSP)
keyword 6-19

receive-turnaround-indication (RCVTRNRND)
keyword 6-19

receiving

cancel signals 6-19

data 3-11

fail signal 6-19

negative response 6-19

record

actual length 5-6

blocked record count 5-6

determining

format of returned 5-16

maximum length 4-5

format name 5-6

length 5-6

selecting formats 5-9

record identifier selection (RECID) keyword

compare values 5-9, 6-9

duplicate compare values 6-9

format 5-9, 6-8

use 5-9, 6-8

records, spooled trace 12-3

related printed information H-1

release considerations 8-4

release operation

considerations 8-4

example 3-13, 6-17, 7-20

language operations A-2

processing 5-19

releasing a program device 5-19

remote format selection (*RMTFMT) parameter 5-10

remote location name (RMTLOCNAME)

parameter 4-11

remote location parameter

ADDICFDEVE command 3-6

OVRICFDEVE command 3-6

remote network id (RMTNETID) parameter 4-12

remote program

considerations 8-5

procedure start request 3-8

start request 3-7

remote system

program start example 3-7

started by application program, example 3-17

starting a program 3-7

target program 5-21

remotely started sessions 3-19

Remove Intersystem Communications Function Device Entry (RMVICFDEVE) command

description 4-2

use 4-2, 4-9

Remove Prestart Job Entry (RMVPJE) command 8-8

request to write with invite (\$\$RCD) system-supplied format 7-6, 7-14

request-to-change transmission direction 6-13

request-to-write function 7-14

request-to-write (RQSWRT) function 6-13

requesting devices

description 3-8

input output following an acquire (ICF file) 8-2

requesting program device, definition 3-8

respond-to-confirm (RSPCONFIRM) keyword 6-12

response indicator keywords 6-18

retail communications, definition 2-3

Retail Industry Programming Support Services (RIPSS) 2-3

Retrieve Data Area (RTVDTAARA) command 8-9

return codes

checking for

detach 3-12

end of transaction 6-19

operation status 5-7

major

description 2-5, 8-1

types 8-1

minor

description 2-5, 8-1

types 8-1

programming considerations 8-1

purpose 2-1

RIPSS

See Retail Industry Programming Support Services (RIPSS)

RPG/400 programming language

display file 11-26, 11-34
error handling 11-2
examples
 coding E-25, E-28
 multiple-session 11-26
 operation 7-14
 set timer function 7-8
 source program 11-6
 target program 11-17
feedback areas 11-2
file transfer E-25, E-28
interface 11-1
introduction 11-1
multiple-session
 source program 11-34
 target program 11-62
parameter list E-4
qualifiers E-4
request-to-write 7-16
use in FTS E-1

S

SDLC

See synchronous data link control (SDLC)

secure from override (SECURE) parameter 4-17

security

error handling 2-5
evoke 6-1
general description 2-5
prestart jobs 8-9
SECURITY keyword 6-2
sending to a remote system 6-2

security (SECURITY) keyword 6-2

selecting record formats 5-9

send function management header then invite

(\$\$SENDFM) system-supplied format 7-4

send with detach (\$\$SENDET) system-supplied format

description 7-4, 7-18
ending communications 7-18
examples

 COBOL/400 program 7-19
 C/400 example 7-19
 RPG/400 example 7-19
 transaction end 7-18

send with end of group (\$\$SENDE) system-supplied format 7-4, 7-12

send with function management header (\$\$SENDNF) system-supplied format 7-4, 7-6

send with invite (\$\$SEND) system-supplied format 7-4, 7-6

send (\$\$SENDNI) system-supplied format 7-4

sending

data 3-11, 6-4, 7-4
negative-response 7-12

sessions

acquiring 3-5, 3-17
description 3-6
end-of-session
 considerations 8-4
 function 7-19
ending
 communications 6-17
 description 3-13
 example 3-12, 3-14
 sessions 3-13
 system-supplied formats 7-20
 using release operation, example 3-13
establishing, example 3-5
general examples 6-1
multiple
 description 3-16, 10-25, 11-26
 example overview 10-2, 11-3
 levels 3-20
 source program 9-10, 10-32
 target program 9-23, 10-61
 transactions 3-14
remotely starting 3-19
source 5-3
starting
 by remote program start request 3-19
 description 3-6
 example 3-14
 sequence diagrams 3-19
 target 5-3

SNA MSRJ (non-ICF communications) 2-4

SNA upline facility (SNUF)

introduction 2-2
VRYCFG command 3-3, 3-14

SNA 3270 program interface, definition 2-2

SNUF

See SNA upline facility (SNUF)

source

program 3-7, 3-17
sessions 5-3
system 5-20

source program

batch transfer 10-5
example 10-32, 10-37

source program, definition 3-7

spooled trace records 12-3

Start Mode (STRMOD) command 3-3

Start Prestart Job (STRPJ) command 8-8

starting remote programs 3-4, 3-7

subdevice (SUBDEV) keyword 6-5

subsystem description

QBASE 8-5
QCMN 8-5

subsystem description, prestart job entry 8-8

support

additional programming 2-6
COBOL programming language 2-6
RPG/400 programming language 2-6

synchronization level specification 6-3
synchronization level (SYNLVL) keyword 6-3
synchronous data link control (SDLC), definition 2-3
system
 operating 1-1
 source 5-20
 target 5-21
system-supplied formats
 additional formats 7-14
 cancel with invite (\$\$CANL) 7-6, 7-10
 cancel (\$\$CANLNI) 7-10
 cancel-invite operation (\$\$CNLINV) 7-17
 communications functions 7-1
 end-of-session function (\$\$EOS) 7-20
 ending transaction 7-18
 evoke with detach (\$\$EVOKET) 7-2, 7-18
 evoke with invite (\$\$EVOK) 7-2
 evoke (\$\$EVOKNI) 7-2
 example target program start 7-2
 fail (\$\$FAIL) 7-8
 general description 7-1
 introduction 2-5
 map to DDS keywords 7-23
 negative response with invite (\$\$NRSP) 7-6
 negative-response with invite (\$\$NRSP) 7-12
 negative-response (\$\$NRSPNI) 7-12
 problem notification formats 7-8
 receiving data 7-6
 request-to-write (\$\$RCD) 7-14
 request-to-write-with-invite (\$\$RCD) 7-6
 send with detach (\$\$SENDET) 7-4, 7-18
 send with end-of-group (\$\$SENDE) 7-4, 7-12
 send with FMH and invite (\$\$SENDFM) 7-6
 send with function-management-header and invite (\$\$SENDFM) 7-4
 send with function-management-header (\$\$SENDNF) 7-4
 send with invite (\$\$SEND) 7-4, 7-6
 send (\$\$SENDNI) 7-4
 sending data example 7-4
 starting on remote system 7-2
 support 7-22
 timer (\$\$TIMER) 7-7
systems network architecture upline facility (SNUF), definition 2-2

T

target
 sessions 5-3
 system 5-21
target program, definition 3-7
target system, definition E-1
timer function
 description 6-8
 RPG/400 example 7-8
timer (TIMER) keyword 6-8
timer (\$\$TIMER) system-supplied format
 COBOL/400 example 7-7

timer (\$\$TIMER) system-supplied format (continued)
 C/400 example 7-7
 format 7-7
 output field format 7-7
 purpose 7-7
 RPG/400 example 7-7
 use 7-7
timing an ICF operation 6-8, 7-7
Trace Intersystem Communications Function (TRCICF) command 12-1
trace records
 database file 12-5
 ending 12-6
 spooled 12-5
 starting 12-1
 stopping 12-2
transactions
 communications 3-8
 description 3-8
 ending
 communications 3-12
 example 6-16
 with COBOL/400 WRITE statement 7-18
 more than one during a session 6-1
 receiving end of transaction 6-19
 starting 3-8

U

using ICF operations 6-1
using Trace Intersystem Communications Function (TRCICF) command 12-1

V

Variable-Buffer-Management (VARBUFMGMT) keyword 6-4
Vary Configuration (VRYCFG) command 3-3
varying on communications configurations 3-3

W

wait file (WAITFILE) parameter 4-6
wait intervals
 responses 5-15
 specifying maximum 5-13
wait record (WAITRCD) parameter
 description 4-6
 use 4-6
 values
 general 4-6
 using read-from-invited-program-devices 4-6
 without using
 read-from-invited-program-devices 4-6
write operations
 language operations A-2
 sending data (COBOL/400 programming language) 7-4

WRITE statement

- COBOL/400 cancel operation example 7-11
- C/400 cancel operation example 7-11
- end-of-session function 7-19, 7-21
- ending
 - sessions 7-20
 - transactions 7-18
- indicating error conditions 7-8
- negative-response operation examples 7-14
- request-to-write 7-14
- request-to-write examples 7-16
- sending
 - a negative-response 7-12
 - data 7-4
- set timer (COBOL/400) function 7-7
- writing**
 - communications applications
 - COBOL/400 language 10-1
 - C/400 language 9-1
 - RPG/400 language 11-1
 - to a program device 5-8

\$\$SEND system-supplied format

See send with invite (\$\$SEND) system-supplied format

\$\$SENDE system-supplied format

See send with end of group (\$\$SENDE) system-supplied format

\$\$SENDET system-supplied format

See send with detach (\$\$SENDET) system-supplied format

\$\$SENDFM system-supplied format

See send function management header then invite (\$\$SENDFM) system-supplied format

\$\$SENDNF system-supplied format

See send with function management header (\$\$SENDNF) system-supplied format

\$\$SENDNI system-supplied format

See send (\$\$SENDNI) system-supplied format

\$\$TIMER system-supplied format

See timer (\$\$TIMER) system-supplied format

***NOMAX value 4-6**

Numerics

3270 device emulation 2-4

3270 program interface

- ICF communications using SNUF 2-2
- non-ICF communications using BSC 2-4

Special Characters

\$\$CANL system-supplied format

See cancel with invite (\$\$CANL) system-supplied format

\$\$CNLINV system-supplied format

See cancel invite (\$\$CNLINV) system-supplied format

See cancel (\$\$CANLNI) system-supplied format

\$\$EOS system-supplied format

See end-of-session (\$\$EOS) system-supplied format

\$\$EVOK system-supplied format

See evoke with invite (\$\$EVOK) system-supplied format

\$\$EVOKET system-supplied format

See evoke with detach (\$\$EVOKET) system-supplied format

\$\$EVOKNI system-supplied format

See evoke (\$\$EVOKNI) system-supplied format

\$\$NRSP system-supplied format

See negative-response-with-invite (\$\$NRSP) system-supplied format

\$\$NRSPNI system-supplied format

See negative-response (\$\$NRSPNI) system-supplied format

\$\$POSRSP system-supplied format

See positive-response (\$\$POSRSP) system-supplied format

\$\$RCD system-supplied format

See request to write with invite (\$\$RCD) system-supplied format

Readers' Comments

**Application System/400
Communications:
Intersystem Communications Function
Programmer's Guide
Version 2**

Publication No. SC41-9590-00

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.



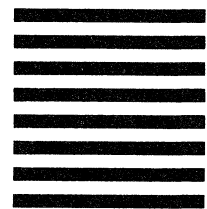
Fold and Tape

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

Readers' Comments

**Application System/400
Communications:
Intersystem Communications Function
Programmer's Guide
Version 2**

Publication No. SC41-9590-00

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.



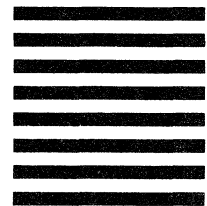
Fold and Tape

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



Program Number: 5738-SS1

Printed in Denmark by
visoprint as

SC41-9590-00

