Compuware Shared Services

User/Reference Guide
MVS Version

Release 8.0

Please direct questions about Compuware Shared Services
or comments on this document to:

Compuware Shared Services Technical Support
Compuware Corporation
One Campus Martius
Detroit, MI 48226-5099
1-800-538-7822

Outside the USA and Canada, please contact
your local Compuware office or agent.

This document and the product referenced in it are subject to the following legends:

Copyright 2003 Compuware Corporation. All rights reserved. Unpublished rights reserved under the Copyright Laws of the United States.

U.S. GOVERNMENT RIGHTS-Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in Compuware Corporation license agreement and as provided in DFARS 227.7202-1(a) and 227.7202-3(a) (1995), DFARS 252.227-7013(c)(1)(ii) (OCT 1988), FAR 12.212(a) (1995), FAR 52.227-19, or FAR 52.227-14 (ALT III), as applicable. Compuware Corporation.

This product contains confidential information and trade secrets of Compuware Corporation. Use, disclosure, or reproduction is prohibited without the prior express written permission of Compuware Corporation. Access is limited to authorized users. Use of this product is subject to the terms and conditions of the user's License Agreement with Compuware Corporation.

Abend-AID, Abend-AID for CICS, CICS Abend-AID/FX, Abend-AID Fault Manager, Abend-AID for WebSphere MQ, XPEDITER, File-AID for DB2, Compuware Shared Services, CSS, Compuware Viewing Facility, Compuware/VF, Enterprise Common Components, and ECC are trademarks or registered trademarks of Compuware Corp.

IBM, CICS, DB2, and MVS/ESA are trademarks or registered trademarks of International Business Machines Corp.

Adobe ® Acrobat ® Reader copyright © 1987-2003 Adobe Systems Incorporated. All rights reserved. Adobe and Acrobat are trademarks of Adobe Systems Incorporated.

All other company and product names are trademarks or registered trademarks of their respective owners.

Doc. CWCXRX8A
December 31, 2003.

Contents

Figures xiii

Tables xvii

Summary of Changes xix
CSS Release 8.0 xix
Support for Earlier Releases xx
CSS Release 7.9 xx

Introduction xxiii
Manual Structure xxiii
Intended Audience xxiv
Related Publications xxiv
FrontLine Support Website xxiv
Online Documentation xxiv
World Wide Web xxv
Notation Conventions xxv
Terminology xxv
Reading the Syntax Diagrams. xxv
Getting Help xxvi

Chapter 1. CSS Overview 1-1
Benefits of CSS 1-1
System Environment 1-2
CSS Components 1-3
Common Files and Utilities 1-3
DDIO Files 1-3
CSS Utilities and Batch File Utilities 1-3
Compuware Viewing Facility (Compuware/VF) 1-3
CSS Language Processors (LP) 1-4
Language Processor Types 1-4
Preprocessor 1-5
Postprocessor 1-6
Security Exit Program 1-6
Executing CSS 1-7
Base Services Overview 1-7
HCI Overview 1-8

Chapter 2. Allocating and Formatting DDIO Files 2-1
Understanding DDIO Files 2-1
Source Listing DDIO Files 2-2
Abend-AID Report DDIO Files 2-2
Abend-AID Report Databases and Shared Directories. 2-2
Transaction Databases and Shared Directories 2-3
Source Listing Databases and Shared Directories 2-3
CSS Utilities 2-3
Batch Utilities 2-4
Source Extraction Utility 2-4
Viewing DDIO File Information 2-4
DDIO Terminology 2-5
Selecting a File Access Method 2-6

DDIO File Considerations 2-7
Sharing Source Listing Files 2-7
Locking Your DDIO Members 2-7
Manual and Automatic Locking 2-7
AUTODELETE Options. 2-8
DDIO File Full Condition 2-8
Resolving a DDIO File Full Condition 2-8
Allocating a DDIO File 2-9
Step 1. Calculate the average member size in KBs. 2-9
Step 2. Calculate the DDIO file size. 2-10
Step 3. Calculate the allocation in blocks.. 2-10
Formatting a DDIO file 2-10
Formatting Formulas 2-11
Step 1. Calculate the total number of blocks in the file. 2-11
Step 2. Calculate the value for the GROUPCOUNT. 2-11
Step 3. Calculate the minimum value for BLKSIZE or CISIZE. 2-11
Understanding DDIO File Limitations 2-12
Allocating and Formatting Example 2-13
Allocating the VSAM Cluster 2-13
Step 1. Determine the average DDIO file member size. 2-13
Step 2. Calculate the DDIO file size. 2-13
Step 3. Calculate the allocation in blocks.. 2-13
Formatting the File 2-13
Step 1. Calculate the total number of blocks in the file. 2-13
Step 2. Calculate the value for the GROUPCOUNT. 2-13
Step 3. Calculate the minimum value for CISIZE. 2-13
Formatting Specifications 2-14
Tuning Your DDIO File 2-15
Tuning the Sample File 2-15
Decreasing CISIZE 2-15
Reducing the Size of DDIO Source Listing Members. 2-16
Using the DIRX Report 2-17
Using the Previous DIRX Report Format 2-17
Source Listing File DIRX Report 2-17
Source Listing File DIRX Report Fields. 2-20
Other Information 2-20
Column Headings 2-21
Member Summary 2-22
Space Summary Based on AUTODELETE. 2-23
Abend-AID Report File DIRX Report 2-24
Abend-AID Report File DIRX Report Fields 2-27
Other Information 2-27
Column Headings 2-28
Member Summary 2-28
Space Summary Based on AUTODELETE. 2-28
Transaction Database DIRX Report 2-30
Transaction Database DIRX Report Fields 2-33
Other Information 2-33
Column Headings 2-34
Member Summary 2-34
Space Summary Based on AUTODELETE. 2-35
Shared Directory DIRX Report 2-36
Shared Directory DIRX Report Fields. 2-47
Other Information 2-47
Source Listing Database DIRX Report 2-49
Source Listing Database DIRX Report Fields 2-52

Other Information 2-52
Column Headings 2-53
Member Summary 2-54
Space Summary Based on AUTODELETE 2-55
Source Listing Shared Directory (SRCEDIR) DIRX Report 2-56
Shared Directory DIRX Report Fields 2-63
Other Information 2-63

Chapter 3. CSS Utilities 3-1
Language Processor Functions 3-1
DDIO File Functions 3-1
Getting Started 3-2
SETTINGS - General Settings 3-3
CREATEJCL - Preparing Compiler Listings for the LP 3-4
Preprocessing 3-5
Running the Preprocessor Using Sample JCL 3-5
Running the Preprocessor Using CSS Utilities 3-5
Selecting Specific Members for Processing 3-7
Submitting Your JCL for Processing 3-7
Postprocessing 3-7
Running the Postprocessor Using Sample JCL 3-8
Running the Postprocessor Using CSS Utilities. 3-8
Selecting Specific Members for Processing 3-10
Submitting Your JCL for Processing 3-10
SOURCE - Working with Source Listing Files 3-10
Setting Source Listing File Options 3-11
Creating and Formatting Source Listing Files 3-12
Setting Advanced Create/Format Parameters 3-15
Deleting Source Listing Files 3-19
Displaying File Information 3-20
Working with Source Listing File Members 3-21
Displaying Members 3-22
Copying Members 3-24
Moving Members 3-26
Exporting Members 3-26
Importing Members 3-27
Selecting Specific Members for Processing 3-28
Setting Advanced Parameters 3-29
Advanced Functions - Source 3-30
Working with Source Listing Files (File Manipulation). 3-30
Working with Source Shared Directory Files (File Manipulation). . . 3-31
Working with Source Listing Database Files (File Manipulation) . . . 3-34
AAREPORT - Working with Abend-AID Report Files 3-37
Setting Abend-AID Report File Options 3-37
Creating and Formatting Abend-AID Report Files 3-38
Setting Advanced Create/Format Parameters 3-42
Deleting Abend-AID Report Files 3-45
Displaying File Information 3-46
Working with Abend-AID Report File Members 3-47
Displaying Members 3-48
Copying Members 3-48
Moving Members 3-50
Exporting Members 3-50
Importing Members 3-51
Selecting Specific Members for Processing 3-52
Setting Advanced Parameters 3-52
Advanced Functions - Abend-AID. 3-54

Working with Abend-AID Report Files (File Manipulation) 3-54
Working with Report Shared Directory Files (File Manipulation) . . 3-55
Working with Abend-AID Report Database Files (File Manipulation) 3-58
Compuware DDIO Calculator — DDIOCALC 3-61
Usage 3-61
Helpful Hints 3-63

Chapter 4. Batch File Utility CWDDSUTL 4-1
Summary of CWDDSUTL Utility Commands 4-1
Usage Considerations 4-2
DD Statements Used with CWDDSUTL. 4-2
Usage Restrictions 4-4
Command Conventions 4-5
Command Parameters for DDIO Files. 4-6
Command Descriptions and Syntax 4-15
COPY 4-15
DELETE (DEL). 4-16
DIRECTORY (DIR, D) and DIRX (DX) 4-17
DIRS 4-18
EXPORT 4-19
FLAG. 4-20
FORMAT Command. 4-23
IMPORT 4-25
LOCK 4-26
MOVE. 4-27
PRINT (P) 4-28
REORDER 4-29
UNLOCK 4-30

Chapter 5. Batch File Utility CWFXSDUT 5-1
Summary of CWFXSDUT Utility Commands 5-1
Usage Considerations 5-3
Renaming Shared Directories and Transaction Databases 5-3
DD Statements Used with CWFXSDUT 5-3
Usage Restrictions 5-4
Command Conventions 5-5
Command Parameters for Shared Directories and Transaction Databases . . . 5-5
Commands Descriptions and Syntax 5-15
ATTACH (ATT) 5-15
CHANGE (CHG) 5-16
COPY 5-17
CREATE (CRE) 5-18
DELETE (DEL) 5-22
DETACH 5-24
DIRECTORY (DIR, D) and DIRX (DX) 5-25
DIRECTORY SEQUENTIAL (DIRS) 5-26
ESTIMATE. 5-27
EXAMINE 5-28
EXPAND 5-30
EXPORT 5-31
IMPORT 5-33
LOCK 5-34
MOVE 5-35
RESET 5-36
SCRATCH (SCR) 5-36
UNLOCK (UNLK) 5-37

VERIFY (VER) 5-38

Chapter 6. Batch File Utility CWAASDUT 6-1
Summary of CWAASDUT Utility Commands 6-2
Usage Considerations 6-3
Renaming Shared Directories and Report Files 6-3
DD Statements Used with CWAASDUT 6-3
Usage Restrictions 6-4
Command Conventions 6-4
Command Parameters for Shared Directories and Report Files 6-5
Commands Descriptions and Syntax 6-14
ATTACH (ATT) 6-14
CHANGE (CHG) 6-15
COPY 6-16
CREATE (CRE) 6-17
DELETE (DEL) 6-20
DETACH 6-22
DIRECTORY (DIR, D) and DIRX (DX) 6-23
DIRECTORY SEQUENTIAL (DIRS) 6-24
ESTIMATE 6-25
EXAMINE 6-26
EXPAND 6-27
EXPORT 6-28
IMPORT 6-30
LOCK 6-31
MOVE 6-32
RESET 6-33
SCRATCH (SCR) 6-34
UNLOCK (UNLK) 6-35
VERIFY (VER) 6-36

Chapter 7. Batch File Utility CWDDLPUT 7-1
Summary of CWDDLPUT Utility Commands 7-1
Usage Considerations 7-3
Renaming Shared Directories and Source Listing Databases 7-3
Dynamic Creation of Source Listing Databases 7-3
DD Statements Used with CWDDLPUT 7-3
Usage Restrictions 7-5
Command Conventions 7-5
Command Parameters for Shared Directories and Source Listing Databases . . 7-6
Command Descriptions and Syntax 7-17
ATTACH (ATT) 7-17
CHANGE (CHG) 7-18
CONVERT (CNVT) 7-20
COPY 7-21
CREATE (CRE) 7-22
DELETE (DEL) 7-27
DETACH 7-30
DIRECTORY (DIR, D) and DIRX (DX) 7-31
DIRECTORY SEQUENTIAL (DIRS) 7-32
ESTIMATE 7-33
EXPAND 7-34
EXPORT 7-35
IMPORT 7-37
LOCK 7-38
MOVE 7-39
PRINT (P) 7-40

RESET 7-41
SCRATCH (SCR) 7-41
TEST 7-42
UNCONVERT (UNCNVT) 7-43
UNLOCK (UNLK) 7-44
VERIFY (VER) 7-45

Chapter 8. Compuware Viewing Facility 8-1
Abend-AID Primary Menu 8-1
Compuware/VF Entry Panel 8-2
Entry Panel Fields. 8-3
Processing Options. 8-5
Source Browsing Options 8-6
Alternate Logon 8-6
Viewing the Release Level 8-6
Parameters Required for Viewing Different Types of DDIO Files 8-7
Creating JCL for LP Preprocessing and Postprocessing 8-7
Creating and Formatting Source Listing Files 8-7
Creating and Formatting Abend-AID Report Files 8-8
Working with Contact Information 8-9
Maintaining Contact Data 8-9
Displaying Contact Data 8-10
Dataset Directory Screen 8-10
Abend-AID Dataset Directory 8-10
Source Listing Dataset Directory 8-12
Primary Commands. 8-16
LOCATE (L) jobname/programname. 8-16
NEXT (N) 8-17
SORT. 8-17
Line Commands 8-18
Hardcopy Options Screen 8-19
Viewing Source Listings 8-21
Viewing Abend-AID Reports 8-21
Viewtime Restriction 8-21
Output Selection Menu for Abend-AID 8-21
Options 0-5. 8-22
Option 6 — Program Storage 8-23
Option 7—Program Listing 8-25
Selecting a Program Listing 8-26
PLIST Operands 8-26
Option 8—Data Management Control Blocks 8-26
Option 9 — IMS Diagnostic Information and PCBs 8-28
Option 10 — CA-IDMS Diagnostic and Status Information 8-28
Option 11 — DB2 Diagnostic and Status Information 8-28
Option 12 — Current COBOL Sort Information 8-28
Option 13 — Creating COBOL Data Locator Reports 8-28
Option 14 — Abend-AID for WebSphere MQ 8-31
Option 15 — LE Information. 8-31
Option 16 — Abend-AID Termination Information 8-31
Mismatched Source Listing Warnings 8-31
Mismatched Date and Time. 8-31
Mismatched Structure for PL/I. 8-32
Mismatched Structure for Assembler. 8-33
On-the-Fly Language Processing Support 8-34
Displaying Available Source Listings 8-38
Line Commands from Source Listing Dataset Directory Screen . . . 8-40
Source-Not-Found Warning 8-41

SOURCE Command 8-41
VIEW (V) Command 8-42
Browsing Abend-AID Reports 8-43
FIND and FS Commands 8-43
Accessing File-AID from Compuware/VF 8-44
Accessing CSS Utilities from Compuware/VF 8-44

Chapter 9. COBOL Language Processor 9-1
Operation 9-1
Preprocessor 9-1
Postprocessor 9-2
Summary of COBOL Language Processor Commands. 9-3
Usage Considerations 9-4
Language Processor DD Statements 9-4
COBOL Samples and Program Names 9-4
Other Usage Considerations 9-5
Output Parameter Considerations 9-6
OUTPUT - DDIO Value 9-6
OUTPUT - PRINT Value 9-7
Performance Considerations 9-7
Usage Restrictions 9-7
Restricted Compiler Options. 9-7
Other Restrictions 9-7
Required Options for Supported COBOL Compilers 9-9
Enhanced Listing 9-10
Listing Features 9-12
COBOL LP Command Descriptions and Syntax 9-13
CICSTEST (CIC) 9-13
COBOL (COB) 9-14
CONDDDIO (CONDD) 9-15
CONDERRL (CONDE) 9-15
CONDSTEP (CONDS) 9-16
DDIO (DD) 9-17
File-Overrides Commands 9-19
INCLUDE (IN) 9-22
LANGPARM (LANGP) 9-23
LANGUAGE (LANG) 9-24
PRINT (PR) 9-26
PROCESSOR (PROC) 9-28
PROGRAM-NAME (PRO) 9-34
SYSTEM (SYS) 9-34

Chapter 10. PL/I Language Processor 10-1
Operation 10-1
Preprocessor 10-1
Postprocessor. 10-2
Summary of PL/I Language Processor Commands. 10-3
Usage Considerations 10-4
Language Processor DD Statements 10-4
PL/I Samples and Program Names 10-4
Other Usage Considerations 10-5
Usage Restrictions and Examples 10-6
Required Options for Supported PL/I Compilers 10-10
PL/I Language Processor Command Descriptions and Syntax 10-11
CONDDDIO (CONDD) 10-11
CONDSTEP (CONDS) 10-11
DDIO 10-12

File-Overrides Commands 10-13
INCLUDE (IN) 10-16
LANGPARM (LANGP) 10-17
LANGUAGE (LANG) 10-17
PL/I 10-18
PRINT (PR) 10-19
PROCESSOR (PROC) 10-20
PROGRAM-NAME (PRO) 10-22
SYSTEM (SYS) 10-22

Chapter 11. Assembler Language Processor 11-1
Operation 11-1
Preprocessor 11-1
Postprocessor 11-2
Summary of Assembler Language Processor Commands 11-3
Usage Considerations 11-3
Language Processor DD Statements 11-4
Assembler Samples and Program Names 11-4
Other Usage Considerations 11-5
Usage Restrictions 11-6
Required Options for Supported Assemblers 11-6
Assembler LP Command Descriptions and Syntax 11-7
ASSEMBLER (ASSEM) Command 11-7
CONDDDIO (CONDD) Command 11-8
CONDSTEP (CONDS) Command 11-8
DDIO Command 11-9
File-Overrides Commands 11-11
INCLUDE (IN) Command 11-14
LANGPARM (LANGP) Command 11-15
LANGUAGE (LANG) Command 11-15
PRINT (PR) Command 11-16
PROCESSOR (PROC) Command 11-17
PROGRAM-NAME (PRO) Command 11-19
SYSTEM (SYS) Command 11-20

Chapter 12. C Language Processor 12-1
Operation 12-1
Preprocessor 12-1
Postprocessor 12-2
Summary of C Language Processor Commands 12-3
Usage Considerations 12-4
DDIO Member Name 12-4
Language Processor DD Statements 12-4
C Samples and Program Names 12-4
Other Usage Considerations 12-5
Usage Restrictions 12-6
Required Options for Supported C Compilers 12-7
C Language Processor Command Descriptions & Syntax 12-8
C 12-8
CONDDDIO (CONDD) 12-9
CONDSTEP (CONDS) 12-9
DDIO 12-10
File-Overrides Commands 12-11
INCLUDE (IN) 12-13
LANGPARM (LANGP) 12-14
LANGUAGE (LANG) 12-14
PRINT (PR) 12-15

PROCESSOR (PROC) 12-16
PROGRAM-NAME (PRO) 12-17
SYSTEM (SYS). 12-17

Chapter 13. CSS Security Exit 13-1
Operation 13-1
Security Exit Program Functions 13-1
Abend-AID for CICS Security Exit Considerations 13-2
Security Exit Program Inputs and Outputs 13-3
Input Information 13-3
Command List 13-3
Output Information 13-6
ADDRESS 13-6

Appendix A. Using the Problem Documentation Utility A-1
Step 1. Add a DD Card to the Language Processor. A-1
Step 2. Run the Language Processor Using Your Source Program A-2
Step 3. Copy Files to Tape. A-3

Appendix B. Using the Customer Modification Facility B-1
Creating a Configuration Source Member B-2
Step 1. Copy member CXCFGINI B-2
Step 2. Edit member CXCFGSET. B-2
Step 3. Submit member CXCFGSET B-2
Reconstructing a Configuration Source Member. B-3
Step 1. Edit Member CXCFGEXT B-3
Step 2. Update Extracted Member. B-3
Step 3. Submit Member CXCFGSET B-3

Appendix C. Using the CWDDUNLP Source Extraction Utility C-1

Appendix D. Using the CWDDCNVA Report File Conversion Utility D-1
Using the CWDDUCNA Report File Unconvert Utility D-1

Appendix E. On-the-Fly Language Processing Support E-1
CXOTFPRE & CXOTFPST Examples. E-1

Glossary G-1

Index. IX-1

Figures

1-1. TP Subsystem, HCI and Application 1-8
2-1. DIRX Report for Allocating DDIO Files 2-10
2-2. Sample DIRX Report for a Source Listing File (*Part 1 of 2*) 2-18
2-3. Sample DIRX Report for a Source Listing File (*Part 2 of 2*) 2-19
2-4. Sample DIRX Report for an Abend-AID Report File (*Part 1 of 2*) 2-25
2-5. Sample DIRX Report for an Abend-AID Report File (*Part 2 of 2*) 2-26
2-6. Sample DIRX Report for a Transaction Database. (*Part 1 of 2*) 2-31
2-7. Sample DIRX Report for a Transaction Database. (*Part 2 of 2*) 2-32
2-8. Sample Portion of DIRX Report for a Shared Directory (*Part 1 of 7*) 2-37
2-9. Sample Portion of DIRX Report for a Shared Directory (*Part 2 of 7*) 2-38
2-10. Sample Portion of DIRX Report for a Shared Directory (*Part 3 of 7*) — Abend-AID for CICS only 2-39
2-11. Sample Portion of DIRX Report for a Shared Directory (*Part 3 of 7*) — Abend-AID for CICS only 2-40
2-12. Sample Portion of DIRX Report for a Shared Directory (*Part 5 of 7*) — Abend-AID for CICS only 2-41
2-13. Sample Portion of DIRX Report for a Shared Directory (*Part 6 of 7*) — Abend-AID for CICS only 2-42
2-14. Sample Portion of DIRX Report for a Shared Directory (*Part 7 of 7*) — Abend-AID for CICS only 2-43
2-15. Sample Portion of DIRX Report for an Abend-AID Shared Directory Member List 2-44
2-16. Sample Portion of DIRX Report for an Abend-AID Database (*Part 1 of 2*) 2-45
2-17. Sample Portion of DIRX Report for an Abend-AID Database (*Part 2 of 2*) 2-46
2-18. Sample DIRX Report for a Source Listing Database. (*Part 1 of 2*) 2-50
2-19. Sample DIRX Report for a Source Listing Database. (*Part 2 of 2*) 2-51
2-20. Sample DIRX Report for a Source Listing Shared Directory. (*Part 1 of 6*) 2-57
2-21. Sample DIRX Report for a Source Listing Shared Directory. (*Part 2 of 6*) 2-58
2-22. Sample DIRX Report for a Source Listing Shared Directory. (*Part 3 of 6*) 2-59
2-23. Sample DIRX Report for a Source Listing Shared Directory. (*Part 4 of 6*) 2-60
2-24. Sample DIRX Report for a Source Listing Shared Directory. (*Part 5 of 6*) 2-61
2-25. Sample DIRX Report for a Source Listing Shared Directory. (*Part 6 of 6*) 2-62
3-1. CSS Utilities - Primary Menu 3-2
3-2. CSS Utilities - General Settings 3-4
3-3. CSS Utilities - CREATEJCL Menu 3-5
3-4. CSS Utilities - Preprocessor Screen 3-6
3-5. CSS Utilities - Postprocessor Screen 3-8
3-6. CSS Utilities - Source Listing File Facility Screen 3-11
3-7. CSS Utilities - Select Desired File Type 3-12
3-8. CSS Utilities - Create/Format Source Listing File Screen (Standard DDIO Files) 3-13
3-9. CSS Utilities - Create/Format Source Listing Shared Directory 3-13
3-10. CSS Utilities - Create Source Listing Database (Attached to a Shared Directory) 3-14
3-11. Source Listing File Create/Format - Advanced Parameters Screen (Sequential) 3-16
3-12. Source Listing File Create/Format - Advanced Parameters Screen (VSAM) 3-16
3-13. Shared Directory Create/Format - Advanced Parameters Screen 3-17
3-14. Source Listing Source Listing Database Create/Format - Advanced Parameters Screen (VSAM) 3-17
3-15. Source Listing Source Listing Database Create/Format - Advanced Parameters Screen (Sequential) 3-18
3-16. Source Listing File Information Report 3-20
3-17. Source Listing Database Information Screen 3-21
3-18. Source Listing Shared Directory Information Screen 3-21
3-19. Source Listing File Member List - DDIO Format 3-22
3-20. Source Listing Shared Directory Member List Screen (Not LONGNAME) 3-23
3-21. Source Listing Shared Directory Member List Screen (LONGNAME) 3-23
3-22. Source Listing Database Member List Screen (Not LONGNAME) 3-24

3-23. Source Listing Database Member List Screen (LONGNAME) 3-24
3-24. CSS Utilities - Source Listing File Move Facility Screen 3-25
3-25. Select Desired File Type Panel 3-28
3-26. Selecting Members for COPY, MOVE, or IMPORT 3-28
3-27. Source Listing File Move Function - Advanced Parameters 3-29
3-28. Source DDIO File - Source Listing Advanced Functions - Main Panel 3-30
3-29. Source DDIO File - CSS Utilities CONFIRM DELETE 3-31
3-30. Source Listing Shared Directory - Advanced Functions - Main Panel 3-31
3-31. Shared Directory - Attached Database Report 3-31
3-32. CSS Utilities - SCRATCH Database 3-32
3-33. CSS Utilities - EXPAND Shared Directory 3-32
3-34. CSS Utilities - Delete Shared Directory 3-33
3-35. CSS Utilities - VERIFY Shared Directory 3-33
3-36. Source Listing Source Listing Database - Advanced Functions - Main Panel 3-34
3-37. CSS Utilities - ATTACH Database 3-34
3-38. CSS Utilities - DETACH Database 3-35
3-39. CSS Utilities - CHANGE AUTODELETE 3-35
3-40. CSS Utilities - DELETE Database 3-36
3-41. CSS Utilities - RESET Database 3-36
3-42. CSS Utilities - Abend-AID Report File Facility Screen 3-37
3-43. CSS Utilities - Select Desired File Type 3-39
3-44. CSS Utilities - Create/Format Abend-AID File Screen (Standard DDIO Files) 3-39
3-45. CSS Utilities - Create/Format Abend-AID Shared Directory Files Screen 3-40
3-46. CSS Utilities - Create/Format Abend-AID Report Database File 3-40
3-47. Abend-AID Report File Create/Format - Advanced Parms (Sequential) 3-42
3-48. Abend-AID Report File Create/Format - Advanced Parms (VSAM) 3-42
3-49. Abend-AID Report File Shared Directory Create - Advanced Parms 3-43
3-50. Abend-AID Report Database Create/Format - Advanced Parms (VSAM) 3-43
3-51. Abend-AID Report Database Create/Format - Advanced Parms (Sequential) 3-44
3-52. Abend-AID Report File Information Report 3-46
3-53. Abend-AID Report Database Information Screen Format 3-47
3-54. Report Shared Directory Information Screen Format 3-47
3-55. Abend-AID Report File Member List 3-48
3-56. CSS Utilities - Abend-AID Report File Move Facility Screen 3-49
3-57. Select Desired File Type 3-51
3-58. Selecting Members for COPY, MOVE, or IMPORT 3-52
3-59. Abend-AID Report File Move Function - Advanced Parameters Screen 3-53
3-60. Standard DDIO Advanced Function Panel - Abend-AID Report File - Main Panel 3-54
3-61. CSS Utilities - CONFIRM Delete 3-54
3-62. Abend-AID Report File Shared Directory - Advanced Functions - Main Panel 3-55
3-63. Shared Directory - Attached Database Report 3-55
3-64. CSS Utilities - SCRATCH Database 3-56
3-65. CSS Utilities - EXPAND Shared Directory 3-56
3-66. CSS Utilities - DELETE Shared Directory 3-57
3-67. CSS Utilities - VERIFY Shared Directory 3-57
3-68. Abend-AID Report File Report Database - Advanced Functions - Main Panel 3-58
3-69. CSS Utilities - ATTACH Database 3-58
3-70. CSS Utilities - DETACH Database 3-59
3-71. CSS Utilities - CHANGE AUTODELETE 3-59
3-72. CSS Utilities - DELETE Database 3-60
3-73. CSS Utilities - RESET Database 3-60
3-74. DDIOCALC Primary Calculations Screen 3-61
3-75. DDIOCALC DIRX Report Data Input Screen 3-62
3-76. Top of the DIRX Report 3-62
3-77. Bottom of the DIRX Report 3-63
3-78. DDIOCALC DIRX Report Data Input Screen 3-63
5-1. Sample EXAMINE Report 5-29
8-1. Abend-AID Primary Menu 8-1
8-2. Compuware/VF Entry Panel 8-3
8-3. Alternate Logon 8-6

8-4. CSS Utilities - CREATEJCL Menu 8-7
8-5. CSS Utilities - Source Listing File Facility 8-8
8-6. CSS Utilities - Abend-AID Report File Facility 8-8
8-7. Contact Information Entries 8-9
8-8. Inserting or Updating a Contact Entry 8-10
8-9. Abend-AID Dataset Directory Screen 8-10
8-10. Abend-AID Dataset Directory Screen for Shared Directory 8-11
8-11. Abend-AID Dataset Directory Screen for Shared Directory after SHOWDB Command 8-12
8-12. Source Listing Standard DDIO Member List Screen 8-12
8-13. Source Listing Shared Directory Member List Screen (Not LONGNAME) 8-12
8-14. Source Listing Shared Directory Member List Screen after SHOWDB Command (Not LONGNAME) 8-13
8-15. Source Listing Shared Directory Member List Screen (LONGNAME) 8-13
8-16. Source Listing Database Member List Screen (Not LONGNAME) 8-14
8-17. Source Listing Database Member List Screen (LONGNAME) 8-14
8-18. Source Listing Shared Directory Member List Screen (After Entering the Line Command “I” (Info)) 8-19
8-19. Source Listing Database Directory Screen (After Entering the Line Command “I” (Info)) 8-19
8-20. Compuware/VF Hardcopy Options Screen 8-20
8-21. Compuware/VF Output Selection Menu for Abend-AID 8-22
8-22. Compuware/VF Program Storage Selection List 8-23
8-23. Compuware/VF Program Listing Selection List 8-25
8-24. File Summary Selection List 8-26
8-25. File-AID/MVS Invoked from Compuware/VF 8-28
8-26. Data Locator Report 8-29
8-27. Default Variable Pattern for a Data Locator Report 8-30
8-28. FINDDATA Source Screen 8-30
8-29. Mismatched Date and Time Warning for COBOL and PL/1 8-32
8-30. Mismatched-Structure Warning for PL/I 8-33
8-31. Assembler Mismatch List 8-33
8-32. Source Not Found Warning Panel 8-34
8-33. Program Date and Time Do Not Match Source Listing 8-35
8-34. Assembler Mismatch Panel 8-35
8-35. Source Listing Retrieval Menu 8-36
8-36. Select Postprocess Procedure Screen 8-36
8-37. CSS Utilities - Load Existing Listing Screen 8-37
8-38. Post-Submission Selection Menu 8-37
8-39. Source Listing Dataset Directory Screen 8-38
8-40. Source-Not-Found Warning 8-41
9-1. Enhanced Listing - Merged DMAP/Data Division Map 9-11
9-2. Enhanced Listing - File Definition and Record Layout for a VSAM File 9-11
9-3. Enhanced Listing - Merged CLIST/OFFSET Information 9-11
13-1. CWASSECD DSECT 13-5
D-1. JCL to Invoke the CWDDCNVA Utility D-1
D-2. JCL to Invoke the CWDDUCNA Report File Unconvert Utility D-2
E-1. Language Postprocessing Example for CA-Endeavor E-2

Tables

2-1. DDIO File Formats 2-1
2-2. Allocating and Formatting DDIO Files 2-6
2-3. Amount of VSAM Space Allocated per Track by CISIZE 2-12
2-4. DDIO File Size Comparison 2-16
3-1. CSS Utilities General Settings 3-4
3-2. Preprocessor Screen Fields 3-6
3-3. Line Commands on Member Selection Screen. 3-7
3-4. Postprocessor Screen Fields 3-8
3-5. Line Commands on Member Selection Screen 3-10
3-6. Source Listing File Options 3-11
3-7. Create/Format Source Listing File Fields 3-14
3-8. Create/Format Source Listing File Advanced Parameters 3-18
3-9. Source Listing File Move Facility Fields 3-25
3-10. Source Listing File Move Function Advanced Parameters 3-29
3-11. Abend-AID Report File Options 3-37
3-12. Create/Format Abend-AID Report File Fields 3-40
3-13. Create/Format Abend-AID Report File Advanced Parameters 3-44
3-14. Abend-AID Report File Move Facility Fields 3-49
3-15. Abend-AID Report File Advanced Parameters 3-53
4-1. DD Statements Used with CWDDSUTL Batch Commands 4-4
4-2. Abend-AID Report File Sort Keys 4-13
4-3. Source Listing File Sort keys 4-13
4-4. Sample Programs for FORMAT Command 4-23
5-1. DD Statements Used with CWFXSDUT Batch Commands 5-3
5-2. Transaction Database Sort keys for Abend-AID for CICS 5-12
5-3. Sample JCL Members for CREATE Command 5-18
6-1. DD Statements Used with CWAASDUT Batch Commands 6-4
6-2. Report File Sort keys for Abend-AID 6-12
6-3. Sample JCL Members for CREATE Command 6-17
7-1. DD Statements Used with CWDDLPUT Batch Commands 7-4
7-2. Source Listing Database Sort Keys for XPEDITER/TSO 7-14
7-3. Sample JCL Members for CREATE Command 7-22
8-1. Abend-AID Primary Menu Options 8-2
8-2. Line Commands on Contact Information Screen 8-9
8-3. Abend-AID Report File Sort Keys 8-17
8-4. Source Listing File Sort keys 8-18
8-5. Accessing File-AID 8-44
9-1. Files Dynamically Allocated by the Preprocessor 9-2
9-2. Language Processor DD Statements. 9-4
9-3. COBOL Sample JCL Members 9-4
9-4. COBOL Program Names 9-5
10-1. Files Dynamically Allocated by the Preprocessor 10-2
10-2. Language Processor DD Statements. 10-4
10-3. PL/I Sample JCL Members 10-4
10-4. PL/I Program Names 10-4
11-1. Files Dynamically Allocated by the Preprocessor 11-2
11-2. Language Processor DD Statements. 11-4
11-3. Assembler Sample JCL Members 11-4
11-4. Assembler Program Names 11-4
12-1. Files Dynamically Allocated by the Preprocessor 12-2
12-2. Language Processor DD Statements. 12-4
12-3. C Sample JCL Members 12-4
12-4. C Program Names 12-5
13-1. Security Exit Enablement for Abend-AID for CICS Commands 13-2
A-1. Problem Documentation Utility Datasets A-2
A-2. Symbolic Parameters A-3

C-1. CWDDUNLP Return Codes C-2

Summary of Changes

This section includes a brief description of the enhancements to CSS 7.9. For those with an earlier CSS release, changes made for CSS 7.8 are also summarized.

CSS Release 8.0

CSS Release 8.0 offers the following new features and enhancements:

- Additional source support for Enterprise PL/I for Abend-AID. External and controlled storage areas are now supported.
- Source support and mismatch capabilities for VisualAge and Enterprise PL/I within Abend-AID for CICS is now supported.
- Additional cursor-selectable fields have been added within the Abend-AID report between the COBOL file-definition ('FD') program storage and the file sections.
- Additional cursor-selectable panel selection capability has been incorporated within the Abend-AID report for all of the supported languages. Highlighted fields have been added in the Call Trace Summary, Program Listing, and Program Storage sections.
- Enhanced support for COBOL BLF cells. FD information is not always available during the time an Abend-AID report is captured. A more informative message is displayed when viewing program storage when a specified file is not open.
- Users will now be able to adjust the display of vertical hex storage to be relative to zero or one. These commands are available in the File section of the Abend-AID report and additionally in the Sort Information section for COBOL programs.
- The Output Selection Menu of an Abend-AID report will now signify which sections are available by highlighting (reverse-video) those options. If a section that is not available for that particular report is selected, a descriptive message will be displayed stating why that section is not available.
- The time used during the CreateMem process during Abend-AID for CICS dump capture time has been reduced thus freeing up system resources more quickly.
- An improved interface for XPEDITER/CICS DDIO processing has been provided to enhance overall Compuware Shared Services product performance and resource management. This includes dynamically allocating/deallocating source listing files.
- Source shared directory functionality for all supported languages can now be utilized by XPEDITER/TSO release 7.3 and XPEDITER/CICS release 7.6.

  Note: XPEDITER/TSO releases 7.0, 7.1, and 7.2 allow full Shared Directory support for C language and PL/I LONGNAME support only. XPEDITER/CICS releases 7.4 and 7.5 as well as XPEDITER/TSO 7.0, 7.1, and 7.2 support the use of source databases for all languages as standalone DDIO files only.

- A DDIO calculator utility is available in this release to compute the optimal DDIO formatting parameters based on various input in the form of a DIRX report or the actual file.
- Abend-AID/XLS, or source support, will now be referred to as simply Abend-AID.

Support for Earlier Releases

- The Viewing Facility has removed support for the CICS Abend-AID product. CICS Abend-AID was the pre-cursor product to CICS Abend-AID/FX, which was renamed Abend-AID for CICS in this release. CICS Abend-AID/FX implemented a Viewing Server for online dump analysis, so support in the Compuware Viewing Facility was no longer needed.

CSS Release 7.9

CSS Release 7.9 offers the following new features and enhancements:

- Shared Directory Support for Abend-AID Reports

  Abend-AID users can now create/attach multiple DDIO report files to a Shared Directory. The Shared Directory support provides users with a single point of access to all of their report repositories.

- Abend-AID Shared Directory Report Routing Rules Support

  Abend-AID reports can be routed to specific subsets of report databases by specifying routing rules (see the *Abend-AID Installation and Customization Guide* for details) when using the Shared Directory support structure. Based on user-supplied routing criteria (for example, Job Name/Job Accounting Info, etc.), abend reports can be directed to specific subsets of report databases to facilitate customer needs. For example, payroll job abends could be routed to a restricted subset of report databases for security reasons.

- Shared Directory Source Support for All Compiler Languages

  All compiler languages can now be used with the Shared Directory to store/retrieve their Source Listings. This enables Abend-AID source support users to specify a single Source Listing Shared Directory name in place of a source DDIO file to retrieve a source listing, or merge a source listing with a report from a pool of source listing databases. The Shared Directory can be defined with a DBMODEL parameter (or it can be added later), which will allow CSS to dynamically allocate a new database file if all others are full.

  Note: CICS Abend-AID/FX currently does not support Source Shared Directory processing. XPEDITER/TSO releases 7.0, 7.1, and 7.2 allow full Shared Directory support for C language and PL/I LONGNAME support only. XPEDITER/CICS releases 7.4 and 7.5 as well as XPEDITER/TSO 7.0, 7.1, and 7.2 support the use of source databases for all languages as standalone DDIO files only.

- Process Compiler Listings On-The-Fly

  When viewing an Abend-AID report through the Viewing Facility, CSS attempts to merge the source listing with the report. If the source listing cannot be found, the listing can be compiled (preprocessor) into a source DDIO file or an existing compiler listing can be post-processed into a source DDIO file. This enhancement is provided by distributing an exit to two CLISTs (or REXX EXECs) so the customer can write a dialog that will allow a listing to be compiled as described above. Sample dialogs have been supplied to assist in dialog development.

- Abend-AID E-Business Name Changed to Abend-AID for WebSphere MQ

  The Abend-AID E-Business product has been renamed to Abend-AID for WebSphere MQ. All references to E-Business have been replaced in the CSS Viewing Facility.

- Source Support for VisualAge PL/I Abend-AID Reports

CSS now provides support for customers who view Abend-AID reports via the CSS Viewing Facility to also view VisualAge PL/I program source merged with the Abend-AID report.

- Source Support for Multiple Assembler Programs on the Calling Chain List in the Viewing Facility

  In previous releases, when two or more Assembler programs were linked together, only the abending program would get source support. With this enhancement, all programs on the calling chain can be viewed with their source merged with the Abend-AID report.

- Revise Printed Abend-AID Report So It Matches the Viewing Facility Report

  The printed Abend-AID report has been modified so the format of the report is in the same order as the Viewing Facility.

- Viewing Facility Updated to Display COBOL PIC and VALUE Information from the Working-Storage Section

  In previous releases, the COBOL PIC and VALUE information were available in the stored source listing member. The Viewing Facility has been enhanced to include this information in both the online and printed reports in addition to the stored member.

- Viewing Facility Hardcopy Print Routines Updated to Show the Report to be Processed on the Print Routing Screen

  Previously, when multiple members were selected for printing from the Viewing Facility, the report (or source listing member) was not identified on the print routing screen. This enhancement will display some identifying information each time the routing screen is redisplayed.

- Link to Viewing Facility From Within SDSF When Viewing an Abend-AID Dump

  This enhancement enables the user to invoke the Viewing Facility and display the Abend-AID report from within SDSF. The linkage mechanism will place the user directly into the proper report without the need to make note of a job name or report number, and without having to navigate the menus. At the conclusion of the Viewing session, the user will be returned to SDSF.

- Distributed Viewing Support for CICS Abend-AID/FX Transaction Dumps

  Abend-AID customers have the ability to view abend reports and source listings from DDIO files that may be on a remotely connected system. This enhancement provides support for using remote DDIO files to CICS Abend-AID/FX users.

- Cursor-selectable Panel Selection From Within the Viewing Facility

  This enhancement enables the user to tab to certain fields in the Viewing Facility panels, press ENTER, and be routed directly to the appropriate screen for more information on that field.

Introduction

CSS (Compuware Shared Services) is an enterprise-wide tool for sharing compiled program information. CSS is enabled by many Compuware products spanning the Fault Management and Interactive Analysis and Debugging product lines. CSS is distributed on the Enterprise Common Components (ECC) tape, along with License Management System (LMS), Base Services, and Host Communications Interface (HCI). Base Services and HCI are required for Abend-AID with Distributed Viewing Support (DVS) and Abend-AID for CICS.

Manual Structure

This manual explains the CSS facilities and provides you with the necessary information to use the COBOL, Assembler, PL/I, and C language processors. It also describes the CSS batch file utilities and contains the following chapters:

Chapter 1, "CSS Overview": Briefly describes the CSS system environment, components, common files and utilities, Base Services, the Compuware Viewing Facility, language processors, and security exit program.

Chapter 2, "Allocating and Formatting DDIO Files": Provides a detailed discussion of allocating and formatting CSS DDIO files. It explains key concepts and considerations for using DDIO files and resolving error messages. It also includes an example of allocating and formatting a file and sample DDIO status reports.

Chapter 3, "CSS Utilities": Describes CSS Utilities — a series of DDIO file functions presented in step-by-step, panel-driven format. Functions supported by CSS Utilities include creation and manipulation of source listing and report files.

Chapter 4, "Batch File Utility CWDDSUTL": Describes CWDDSUTL, which is used by Abend-AID, Abend-AID for CICS, XPEDITER/CICS, and XPEDITER/TSO. This chapter discusses how to print, delete, lock, unlock, copy, move, import, and export DDIO file members, list DDIO file directory entries, and initially format a DDIO file.

Chapter 5, "Batch File Utility CWFXSDUT": Describes CWFXSDUT, which is used by Abend-AID for CICS. This chapter discusses how to manage the files used by a shared directory and its associated transaction databases.

Chapter 6, "Batch File Utility CWAASDUT": Describes CWAASDUT, which is used by Abend-AID. This chapter discusses how to manage the files used by a shared directory and its associated report files.

Chapter 7, "Batch File Utility CWDDLPUT": Describes CWDDLPUT, which must be used for C Language, VisualAge PL/I, and Enterprise PL/I processing if using LONGNAME compiler support as well as by Abend-AID if using the shared directory for source listings. This chapter discusses how to manage a shared directory (SRCEDIR) and its associated source listing databases.

Chapter 8, "Compuware Viewing Facility": Describes Compuware/VF and how to view, print, lock, unlock, and delete members in DDIO files online.

Chapter 9, "COBOL Language Processor": Describes the outputs and commands associated with the COBOL language processor in order to write a source listing or enhanced listing to a VSAM or sequential DDIO file.

Chapter 10, "PL/I Language Processor": Describes the outputs and commands associated with the PL/I language processor (including VisualAge PL/I and Enterprise PL/I) in order to write a source listing to a VSAM or sequential DDIO file.

Chapter 11, "Assembler Language Processor": Describes the outputs and commands associated with the Assembler language processor in order to write a source listing to a VSAM or sequential DDIO file.

Chapter 12, "C Language Processor": Describes the outputs and commands associated with the C language processor in order to write a source listing to a VSAM or sequential DDIO file.

Chapter 13, "CSS Security Exit": Describes how to prepare the optional CSS Security Exit program that can be used with your existing security package.

Appendix A, "Using the Problem Documentation Utility": Describes how to collect documentation to send to Compuware Technical Support for resolving problems encountered when programs are compiled. This utility helps ensure that all required documentation is collected.

Appendix B, "Using the Customer Modification Facility": Describes how to retrofit restricted zaps from prior releases to CSS 7.4 and more current for the SMP/E environment.

Appendix C, "Using the CWDDUNLP Source Extraction Utility": Describes how to use the CWDDUNLP utility to extract source from a DDIO source listing file.

Appendix D, "Using the CWDDCNVA Report File Conversion Utility": Describes how to use the CWDDCNVA Report File Conversion Utility to migrate a report file to the report database format.

Appendix E, "On-the-Fly Language Processing Support": Describes how to use the Viewing Facility when no source or mismatch situations occur.

Intended Audience

This manual is intended for CSS installers and application programmers. It should only be used with the MVS/ESA, OS/390, and z/OS versions of Compuware products.

Related Publications

For more information about CSS, LMS, or HCI, refer to the Compuware documentation shipped with the ECC tape. For more information about Distributed Viewing Support, refer to the *Abend-AID Installation and Customization Guide* or the *Abend-AID for CICS Installation and Customization Guide.* Instructions for accessing this documentation are given in the following paragraphs. For more information on compiling COBOL, PL/I, Assembler, or C/C++ programs, refer to the appropriate IBM manuals.

FrontLine Support Website

Access online technical support for Compuware products via our FrontLine support website. View or download documentation, frequently asked questions, and product fixes, or directly e-mail Compuware with questions or comments. To access FrontLine, you must first register and obtain a password at **http://frontline.compuware.com**.

Online Documentation

Documentation for this product is provided on CD-ROM in several electronic formats.

- PDF files can be viewed with the free Adobe Acrobat Reader, available at **http://www.adobe.com**.
- HTML files can be viewed with any standard Web browser.
- BookManager softcopy files can be viewed with any version of IBM BookManager READ or the IBM Softcopy Reader. To learn more about BookManager or download the free Softcopy Reader, go to **http://www.ibm.com**.

World Wide Web

Compuware's site on the World Wide Web provides information about Compuware and its products. The address is **http://www.compuware.com**.

Notation Conventions

This section describes common terms and syntax conventions used in this manual. A change bar (|) indicates an update to the manual for this release.

Terminology

The following terms are defined as they are used in this manual:

- **DDIO (Dump Dataset Input Output) file** is used generically to mean any of the following types of files:
  - Report File
  - Source Listing File
  - Transaction Databases & Shared Directories
  - Report Shared Directory & Database
  - Source Listing Shared Directory
  - Source Listing Database

  For a list of the types of DDIO files created by the CSS Utilities and which products use/require them, see Chapter 2, "Allocating and Formatting DDIO Files".
- **Member** is used to identify an individual diagnostic report in an Abend-AID report file, a source listing in a source listing file, or a profile in a profile file.
- **Entry** is used to identify an individual transaction diagnostic report in a transaction database for Abend-AID for CICS.

Reading the Syntax Diagrams

Syntax diagrams define primary command syntax.

A *parameter* is either a keyword or a variable.

- All KEYWORDs are shown in uppercase characters and must be spelled exactly as shown. You cannot substitute another value. If any part of a KEYWORD is shown in lowercase characters, that part is optional.
- Variables are user-specified values and are printed in lowercase italics. For example, *dataset-name* indicates you are to substitute a value.

The syntax for commands is described in diagrams that help you visualize parameter use. The following example shows a command and a parameter:

```
►►─COMMAND──parameter──►◄
```

Read the diagrams from left to right and from top to bottom. These symbols help you follow the path of the syntax:

- ▶▶ indicates the beginning of a statement.
- → indicates the statement is continued on the next line.
- ▶ indicates the statement is continued from the previous line.
- →◀ indicates the end of a statement.

Required parameters appear on the horizontal line (the main path). Optional parameters appear below the main path. Default parameters appear above the main path and are optional. The command executes the same regardless whether the default parameter is included.

```
                                  ┌DEFAULT-KEYWORD─┐
▶▶─COMMAND─REQUIRED-KEYWORD─┴──────────────────┴──┬──────────────────┬─▶◀
                                                  └optional-variable─┘
```

Vertically stacked parameters are mutually exclusive. If you must choose a parameter, one item of the stack appears on the main path. If the parameters are optional, the entire stack appears below the main path. If a parameter in a stack is the default, it appears above the main path.

```
                ┌DEFAULT-KEYWORD1─┐  ┌default-var1─┐
▶▶─COMMAND─┼─────────────────────┼─┼─────────────┼─▶◀
                ├OPTIONAL-KEYWORD2─┤  ├optional-var2─┤
                └OPTIONAL-KEYWORD3─┘  └optional-var3─┘
```

If the same parameters are used with several commands their syntax may be documented in a separate diagram. In the command syntax, these common parameters are indicated with separators before and after the parameter name.

```
▶▶─COMMAND─┤common-parameter├─▶◀
```

An arrow returning to the left indicates a repeatable item. If the arrow contains a comma, separate the repeated items with a comma.

```
                ┌,──────────────┐
▶▶─COMMAND─┴─required-var─┴─▶◀
```

Getting Help

Compuware continually strives to improve our software products and documentation. Feedback from our customers helps us to maintain our quality standards.

While the software should execute according to documented specifications, there are times when problems do occur. If problems arise, check the Compuware manuals for assistance. You may also need to consult your site's technical representative for Compuware assistance.

If you have attempted to solve the problem using the resources listed above and the difficulty persists, contact the Compuware Technical Support Hotline for assistance.

Technical support is also available via our FrontLine Support web site as described in "Related Publications".

Please have the following information available before calling. This information will assist the Technical Support personnel in determining the exact cause of your problem in a timely manner.

Note: If you encounter a CSS problem when compiling your program, Compuware Technical Support staff may need additional information. You can obtain this information by using the CSS Problem Documentation Utility. See the "Using the Problem Documentation Utility" appendix for instructions on using this utility.

1. Use the following sample JCL members to identify the PTFs and/or APARs that have been applied to the respective ECC components' libraries:
   - CXLPTFS or CXLZAPS in the CSS SLCXCNTL library
   - $LMLPTFS in the LMS SLMSCNTL library
   - $DVLPTFS in the Base Services SKMPCNTL library
   - $HCLPTFS in the HCI SLHCCNTL library

   A lengthy SMP/E report will be generated with exceptions to the CXLZAPS job. That will produce a summarized version of the PTF listing.
2. Determine the release number of the ECC-enabled product.
3. For CSS problems, identify the compiler language and the release number being processed.
4. Identify the CICS release in use if the problem involves a CICS product.
5. Identify the operating system release to help determine operating system dependencies.
6. If an abend occurs, note the displacement and the module in which the interrupt occurs.
7. Note the sequence of issued commands that resulted in the abend, the data type, and the programming language being used.
8. Locate your Compuware customer number.

Compuware Technical Support
Compuware Corporation
One Campus Martius
Detroit, MI 48226-5099
1-800-538-7822

http://frontline.compuware.com

Chapter 1.
CSS Overview

Compuware Shared Services (CSS) is an integral component of many Compuware products spanning the Fault Diagnosis, Interactive Analysis and Debugging, and File and Data Management product lines. CSS provides storage, retrieval, and maintenance for Abend-AID reports, source listings, and transaction reports on datasets called DDIO files. It also provides language processing support for COBOL, PL/I, Assembler, and C for OS/390 (also C for MVS/ESA V3 R2).

The following products currently use CSS.

- Abend-AID (VisualAge PL/I and Enterprise PL/I language support is also available)
- Abend-AID Fault Manager
- Abend-AID for WebSphere MQ
- Abend-AID for CICS
- STROBE
- XPEDITER/CICS (VisualAge PL/I and Enterprise PL/I language support is also available)
- XPEDITER/Code Coverage
- XPEDITER/TSO and XPEDITER/IMS (VisualAge PL/I and Enterprise PL/I language support is also available)
-

Benefits of CSS

Prior to CSS Release 1.0, the components that make up CSS were distributed within each product library. For customers with multiple Compuware products, the result was a high degree of redundancy within their product libraries and a complex system of product maintenance. By combining components used by several products into a single unit, CSS provides:

- **Easier Installation**

  All CSS components can be installed at one time rather than piecemeal during installation of other Compuware products.

- **Simplified Maintenance**

  All maintenance to CSS components is applied to ONE set of libraries. It is no longer necessary to search through each product library, determine if a fix must be applied, and then apply the same fix to multiple libraries.

- **Frequent New Release Updates**

  New release updates are made available more frequently.

- **Disk Storage**

  CSS programs and subroutines reside in a separate library. Compuware customers with multiple products installed can recover disk space previously used by duplicate modules and programs using CSS.

- **Fewer JCL Changes**

  The need to change CICS startup JCL, TSO logon PROCs, compile PROCs, and/or rundecks is reduced or eliminated because a central library exists for all products using CSS. This helps to simplify installation.

- **Separate Documentation**

  Documentation for common components is no longer imbedded in the individual product manuals. In addition to your product documentation, you also receive the *Enterprise Common Components Installation and Customization Guide* in printed form, and the *Compuware Shared Services User/Reference Guide* and *Enterprise Common Components Messages and Codes* on CD. You no longer need to hunt through the product manuals for information on formatting DDIO files, printing reports and listings, and setting the parameters to run the language processors. Additionally, this ECC documentation set provides greater detail on processing options and customization than was previously available.

System Environment

CSS supports CICS Releases 1.7 through 4.1. It also supports CICS Transaction Server for OS/390 Release 1.3 and earlier and all releases of IBM MVS, as well as ISPF/PDF Versions 2.0 or more current.

The following is a list of compilers supported by the language processors:

- Assembler
  - High Level (HLASM) Assembler 1.1, 1.2, 1.3, 1.4
  - Assembler H
  - Assembler F
- COBOL
  - Enterprise COBOL for z/OS and OS/390 Release 3.1.0
  - COBOL for OS/390 and VM 2.1, 2.1.1, and 2.2 (MLE Support)
  - COBOL for MVS and VM 1.2 and 1.2.2 (MLE Support)
  - AD/Cycle COBOL/370 1.0, 1.1
  - VS/COBOL II 1.2, 1.3, 1.4
  - OS/VS COBOL 2.4
  - COBOL for z/OS 3.2

    Note: Compuware supports the MVS portion of the IBM compilers COBOL for MVS and VM, and COBOL for OS/390 and VM.

  - CA-OPTIMIZERs
    - CA-OPTIMIZER II 1.1, 1.2, 1.3, 2.0, 2.1, 2.2, 3.0
    - CA-OPTIMIZER III 5.0, 5.1, 6.0, 6.1, 7.0

    Note: CA-OPTIMIZER II Release 2.0 supports COBOL/370.

- PL/I
  - Enterprise PL/I R3.1
  - VisualAge PL/I R2.2
  - PL/I for MVS and VM 1.1.1
  - AD/Cycle PL/I 1.1
  - PL/I 1.5.1, 2.2, 2.3, 3.2

  Note: Compuware supports the MVS portion of the IBM compilers PL/I for MVS and VM 1.1.1.

- C
  - C for MVS/ESA V3 R2
  - OS/390 C releases through V2 R10

CSS Components

CSS consists of the following components:

- Compuware common files and utilities
  - DDIO files, Shared Directories and associated Report, Transaction, and Source Listing Databases
  - Batch file utilities: CWDDSUTL, CWFXSDUT, CWAASDUT, and CWDDLPUT
- Compuware Viewing Facility (Compuware/VF)
- Compuware Language Processors (COBOL, Assembler, PL/I, C, VisualAge PL/I, and Enterprise PL/I)
- Security Exit Program

Common Files and Utilities

CSS components include common files and utilities that provide storage, retrieval, and maintenance functions.

DDIO Files

DDIO is a proprietary file access method that stores Abend-AID reports, transaction reports, and source listings for several Compuware products. You can allocate DDIO files as either VSAM or sequential datasets. Different products may require different types of DDIO files.

For detailed information on allocating and formatting DDIO files, see **Chapter 2, "Allocating and Formatting DDIO Files"**."

CSS Utilities and Batch File Utilities

CSS Utilities walks you step-by-step through CSS language processor functions and DDIO file manipulation using easy-to-use panels.

The batch file utilities (CWDDSUTL, CWAASDUT, and CWDDLPUT) provide, in a batch environment, the same functions as CSS Utilities. The batch file utility CWFXSDUT provides these functions for Abend-AID for CICS files.

For detailed information on these utilities, see **Chapter 3, "CSS Utilities"**, **Chapter 4, "Batch File Utility CWDDSUTL"**, **Chapter 5, "Batch File Utility CWFXSDUT"**, **Chapter 6, "Batch File Utility CWAASDUT"**, and **Chapter 7, "Batch File Utility CWDDLPUT"**.

Compuware Viewing Facility (Compuware/VF)

The Compuware Viewing Facility (Compuware/VF) provides immediate, online, menu-driven access to reports and source listings. From Compuware/VF, you can view a directory that summarizes the members in a DDIO file. You can then access the source listing or Abend-AID report from the directory. While viewing an Abend-AID report, you can directly access any available section of the report, search for particular date or currency formats within the report, or link to Compuware File-AID (to use the BROWSE or EDIT functions only).

Note: To access File-AID from Compuware/VF, one of the following File-AID releases must be installed: File-AID/MVS 8.5 or more current, File-AID for DB2 V3R9 or more current, or File-AID for IMS 4.7 with product update F1. In addition to one of the File-AID products, Abend-AID 9.1 or more current must also be installed.

Compuware/VF has many features that enable you to view DDIO files easily and quickly. Its navigation commands let you directly access desired report sections to eliminate cumbersome scrolling. From Compuware/VF you can view, print, lock, unlock, and delete members in a DDIO file online. The members can easily be manually locked to ensure that they remain stored on the DDIO file until you unlock them. Compuware/VF also lets you print members at view time.

Compuware/VF enables you to access the following types of DDIO files:

- Abend-AID reports
- Abend-AID, Abend-AID for CICS, XPEDITER/TSO, and XPEDITER/CICS source listings.

Note: Viewing access to Abend-AID for CICS source listing files is also provided by the Abend-AID for CICS view server. Abend-AID for CICS users do not need to install Compuware/VF if a server has already been configured to support viewing access. Compuware/VF may be installed, however, in order to view other products' source listing files.

For more information on the Viewing Facility, refer to Chapter 7, "Compuware Viewing Facility."

CSS Language Processors (LP)

The language processors capture information about a compiler listing and store it in a DDIO file. The majority of this information is gathered from the compiler listing, but optionally, SYSIN and SYSLIB are also examined.

A language processor can be run using two methods:

- preprocessor
- postprocessor

You may use the preprocessor or postprocessor (whichever is best for your site as needed.) See the applicable language processor chapter for when to use the pre- or postprocessor.

Language Processor Types

CSS provides language processing support for the following languages:

- COBOL
- PL/I, VisualAge PL/I, Enterprise PL/I
- Assembler
- C

The COBOL language processor reads the compiler listing produced by the COBOL compiler and writes either a compiler listing or an enhanced listing to the DDIO file. The enhanced listing merges the information normally found in the data division map (DMAP) and the condensed listing (CLIST) or procedure map (PMAP) into the source statement lines. The COBOL compiler options in effect are sorted alphabetically and appear at the beginning of the enhanced listing. The enhanced listing provides you with a condensed hardcopy of the compiler information.

The COBOL and PL/I listings may be used to provide source support during a debugging or diagnostic session using any of the following products:

- Abend-AID
- Abend-AID for CICS
- XPEDITER/CICS
- XPEDITER/TSO

The Assembler listings may be used to provide source support during a debugging or diagnostic session using any of the following products:

- Abend-AID
- XPEDITER/CICS
- XPEDITER/TSO

The C language processor reads the compiler listing produced by the C compiler and writes the compiler listing to the DDIO file and SYSCPRT. The source listing in DDIO can then be used to provide source support during a debugging or diagnostic session using:

- XPEDITER/TSO

Language support for Enterprise PL/I and VisualAge PL/I is provided for the following products:

- Abend-AID
- XPEDITER/CICS
- XPEDITER/TSO

Preprocessor

Some information about a program module is not always available from the compiler listing. The preprocessor, in addition to gathering information from the compiler listing, gathers information from SYSIN and SYSLIB. The preprocessor provides additional functionality for XPEDITER/CICS and XPEDITER/TSO, and is required for PL/I programs that use the %NOPRINT statement.

When setting up the language processors, Compuware recommends that you use the preprocessor instead of the postprocessor. Only use the postprocessor in situations where compiled source listings have been stored.

Preprocessor Functions

- Determines the proper compiler options required to process the listing file.
- Automatically invokes the compiler or assembler to compile or assemble your source program.
- Writes the listing to the DDIO file. You can produce an enhanced listing for COBOL programs, if desired. PL/I and Assembler listings are written to SYSPRINT. C listings are written to SYSCPRT.

Preprocessor Benefits

- **Better handling of compiler errors.** By using the preprocessor, you can avoid the potential problem of processing a listing with compiler errors. The preprocessor, in conjunction with the CONDDDIO parameter, internally checks the return code from the compiler and doesn't write a DDIO listing that contains errors. For example, if you set the CONDDDIO parameter to 8, the preprocessor will write the compiler listing to the DDIO file unless the compiler return code exceeds 8. The preprocessor can process the compiler errors more effectively than the postprocessor. Compuware recommends that you use the preprocessor rather than the postprocessor.
- **Capturing suppressed source code.** When you use any of the following parameters, you can suppress sections of source code from the compiler listing:

For COBOL	COPY SUPPRESS
For PL/I	%NOPRINT
For Assembler	PRINT OFF or PRINT NOGEN
For C	NOSHOWINC or NOEXPMAC

The preprocessor captures this information from SYSIN and SYSLIB, or by forcing on more revealing compiler options, then altering the listing to emulate the options you requested.

- **Automated compiler options.** The postprocessor requires that certain compiler options be specified in order to process all needed sections of the compiler listing. The preprocessor can automatically pass the required options to the compiler.
- **Simplified JCL.** While the postprocessor requires that you add a step after the compile step, the preprocessor only requires that you make minor modifications to your existing compile step.

Postprocessor

The postprocessor is executed as a single step and it can be a separate job. It reads in the listing created from the compiler. Information is gathered from the source listing, XREF, data maps, and object code sections of the listing. This information is stored in a DDIO file member and is used by various Compuware products.

While the preprocessor is the preferred method of loading the compiled listing to the DDIO, there are situations that necessitate the use of the postprocessor as a viable alternative. This is commonly found in production environments where the listings may be archived. In order for the listings to populate the DDIO, they must be compiled using the required options specified in the appropriate language processor chapter. Note that these are usually the default options at many shops.

If the programs contain suppressed source code, certain additional control statements must be added to the language processor. Please view the Language Processor chapters for the programming language desired for more details.

If the programs are compiled with OFFSET as well as OPT(imize), certain additional control statements must be added to the language processor. Please view the Language Processor chapters for the programming language desired for more details.

Postprocessor Benefits

- Exact match of compiled output listings with the executable load module. This is especially important in production environments where time is critical.
- Eliminates the risk of copybook or source code changes prior to recompiling the code since the listing reflects the code that is being executed.
- Significantly less time is needed to process source listings as opposed to recompiling the source code.

Security Exit Program

CSS also includes the Compuware Security Exit Program. This is an optional user-written exit for your Compuware products. It can be used in conjunction with your existing security package to secure sensitive data used by the following Compuware products:

- Abend-AID (VisualAge PL/I and Enterprise PL/I language support is also available)
- Abend-AID Fault Manager
- Abend-AID for WebSphere MQ
- Abend-AID for CICS
- STROBE
- XPEDITER/CICS (VisualAge PL/I and Enterprise PL/I language support is also available)
- XPEDITER/Code Coverage
- XPEDITER/TSO and XPEDITER/IMS (VisualAge PL/I and Enterprise PL/I language support is also available)

The Compuware Security Exit enables you to control access to DDIO file members or restrict access to various commands through a user-written exit program. For the Abend-AID for CICS product, you can restrict access to some of the CWFXSDUT functions for transaction database and source listing files. For transaction databases, you can use the Security Exit Program to restrict access to individual dumps within a transaction report database.

If you install a Security Exit program, it will be called by all Compuware products installed at your site. For more information on the Security Exit Program, see Chapter 12, "CSS Security Exit."

Executing CSS

CSS is executed at the following times:

- Abend-AID and Abend-AID for CICS
  - at compile time (when source support is used)
  - at time of abend or application program failure
  - at view time when using Compuware/VF or the Abend-AID for CICS Viewer
  - at installation and maintenance time when formatting a DDIO file.
  - when a compile listing that has been stored elsewhere is needed for Abend-AID or Abend-AID for CICS viewing with source support.
- XPEDITER/TSO and XPEDITER/CICS
  - at compile time (when source support is used)
  - at execution time for interactive debugging
  - at view time when using Compuware/VF
  - at installation and maintenance time when formatting a DDIO file.

Base Services Overview

Base Services and HCI are required to use Abend-AID Distributed Viewing Support (DVS). Abend-AID or Abend-AID for CICS users can use DVS to view reports whether residing on a local or remote system. Base Services is also used by Abend-AID for CICS during regular viewing.

Abend-AID users can view both merged and base reports where either the base report, source listings, or both reside on a remote system. Access to the remote files is provided through the optional Distributed Viewing Support component.

With Distributed Viewing Support, Abend-AID users can utilize source support across remote MVS systems that do not share DASD. Working from individual local systems, any number of users can access report and source listing files that reside locally and/or remotely on other MVS images.

Distributed Viewing Support enables you to access the following types of DDIO files:

- Abend-AID reports
- Abend-AID for CICS dump reports
- Abend-AID, Abend-AID for CICS, XPEDITER/TSO, and XPEDITER/CICS source listings.

Base Services and HCI are distributed with CSS on the Enterprise Common Components distribution tape. Instructions for configuring Distributed Viewing Support and setting up the servers are provided in the Abend-AID and Abend-AID for CICS product documentation. Instructions for installing Base Services and HCI can be found in the *Enterprise Common Components Installation and Customization Guide*.

Distributed Viewing Support is available beginning with Abend-AID Release 9.0 and CICS Abend-AID/FX (now Abend-AID for CICS) release 4.4.

Note: Viewing access to Abend-AID for CICS source listing files is also provided by the Abend-AID for CICS **view** server. Abend-AID for CICS users do not need to install Compuware/VF if a server has already been configured to support viewing access. Compuware/VF may be installed, however, in order to view other products' source listing files.

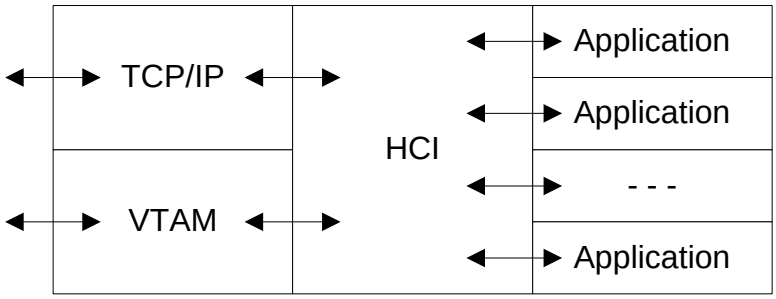
HCI Overview

The Host Communications Interface (HCI) is a facility that provides connectivity between mainframe-based programmer productivity software and peer-node software running on other platforms in a network. These communications facilities provide a simple application programming interface so that Compuware software can be written to communicate with peer programs running on different platforms. Additionally, the communications system is flexible, simple to operate, and reliable. It has been designed to support open-ended communications protocols, and it can operate on older releases of the MVS operating system. It allows application programs to communicate over any one of several protocols without knowing which protocol is in use at any given time.

The HCI also serves as a server activation facility that is responsible for starting and monitoring application programs that are using the communications facilities.

And finally, the HCI has been designed with no architected upper limit on the number of concurrent sessions supported, nor on throughput rates. The following figure shows how the HCI fits into the scheme of the TP application and the MVS subsystem components.

Figure 1-1. TP Subsystem, HCI and Application. HCI Position as Middleware



For more information about the Host Communication Interface features, refer to Chapter 8, "HCI Facilities" in the *Enterprise Common Components Installation and Customization Guide*.

Chapter 2.
Allocating and Formatting DDIO Files

This chapter describes allocating and formatting CSS DDIO (Dump Dataset Input Output) files. It explains key concepts and considerations for using DDIO files and resolving error messages. It also includes an example of allocating and formatting a file and sample DDIO status reports.

CSS DDIO files are used to store diagnostic reports, transaction reports, and source listings. The term *DDIO file* is a generic term used to refer to datasets that store the reports and listings from Compuware products that use CSS.

Understanding DDIO Files

CAUTION:

CSS Utilities and the CWDDSUTL, CWFXSDUT, CWAASDUT, and CWDDLPUT batch file utilities are the only utilities recommended for use with Compuware DDIO files.

File-AID/MVS and other VSAM-related vendor products (including vendor packages that compress or enhance buffer management on VSAM files) should NOT be used against Compuware DDIO files. Doing so can corrupt the contents of DDIO files.

DDIO is a proprietary file access method that stores Abend-AID reports, transaction reports, and source listings for several Compuware products. You can allocate DDIO files as either VSAM or sequential datasets.

Note: Shared directories must be VSAM.

Different products may require different types of DDIO files.

The Compuware Language Processors are a set of programs that analyze the output from a compiler execution and store that information in a source listing DDIO file. Abend-AID and Abend-AID for CICS use source listings when Abend-AID reports or CICS transaction abend reports are formatted with source support. XPEDITER/CICS and XPEDITER/TSO use source listings during interactive debugging. Usually, one or more source listing files can be shared between all Compuware products that use CSS while other DDIO files such as report files, transaction databases, and profile files cannot.

The following table lists the types of DDIO files created by the CSS utilities and which products use/require them:

Table 2-1. DDIO File Formats

Compuware Product	Report File	Source Listing File	Transaction Databases & Shared Dir.	Report Shared Dir. & Database	Source Listing Shared Dir.	Source Listing Database
Abend-AID	x	x		x	x	x
Abend-AID for CICS		x	x		x	x
XPEDITER/TSO		x			x[1] See Note	x

Table 2-1. DDIO File Formats

Compuware Product	Report File	Source Listing File	Transaction Databases & Shared Dir.	Report Shared Dir. & Database	Source Listing Shared Dir.	Source Listing Database
XPEDITER/IMS		x			x[1] See Note	x
XPEDITER/CICS		x			See Note	x

[1] If using XPEDITER/TSO & XPEDITER/IMS with the C Language Processor, VisualAge PL/I, or Enterprise PL/I, and you use Long Program Names, the Source Listing Shared Directory must be used.

Note: XPEDITER/TSO Release 7.3, XPEDITER/CICS Release 7.6 and Abend-AID for CICS Release 4.5 support source listing shared directories. Prior releases of these products **do not** support source listing shared directories. However, individual source listing databases **can** be used by all CSS-enabled Compuware products if the databases are explicitly referenced. See footnote 1 above for exceptions.

Source Listing DDIO Files

Source listing files are DDIO files that contain one or more source listing members. The source listing member contains the compiler listing and other program information collected by the appropriate Compuware Language Processor. Any source listing DDIO file of any type (sequential or VSAM) can be shared among any of the Compuware Products that provide source listing support. For performance reasons, VSAM source listing DDIOs are recommended when using XPEDITER/CICS, and Abend-AID for CICS.

Abend-AID, Abend-AID for CICS, XPEDITER/CICS, and XPEDITER/TSO can use COBOL and PL/I listings.

Abend-AID, XPEDITER/CICS, and XPEDITER/TSO can use Assembler listings.

XPEDITER/TSO also supports C language, VisualAge PL/I, and Enterprise PL/I programs. Short named C language, VisualAge PL/I, or Enterprise PL/I programs (eight character and shorter program names) can be stored and used from conventional source listing DDIO files. See “Source Listing Databases and Shared Directories” on page 2-3 for special source listing DDIO files required for C language, VisualAge PL/I, and Enterprise PL/I programs if using LONGNAME (program names longer than eight characters) compiler support.

Abend-AID Report DDIO Files

Abend-AID creates diagnostic reports at the time of a program fault. It extracts data from the affected application and associated control blocks and writes this data to a DDIO report file. At view time, the data stored in the Abend-AID report is retrieved and formatted into a readable report that can then be displayed online and/or printed.

Abend-AID Report Databases and Shared Directories

Abend-AID diagnostic reports can now be routed into a database pool, which is managed via a shared directory. Abend reports are stored in a report database. Multiple report databases can be attached to a single shared directory and can be managed as a pool of report databases. The Abend-AID install process creates a CWROUTE module which points to the site default shared directory (see the *Abend-AID Installation and Customization Guide* for more information on CWROUTE). However, an alternative shared directory can be specified at abend time by using the ABNLTERM DD statement to point to the desired shared directory.

Transaction Databases and Shared Directories

Abend-AID for CICS creates transaction reports from CICS transaction terminations. It writes extracted data to a transaction database. The transaction databases are managed via a shared directory, which is a special type of VSAM DDIO file. Multiple transaction reports are stored in a transaction databases and multiple transaction databases can be attached to a single shared directory. The shared directory can give you a single, enterprise-wide view of all CICS transaction and region faults. Abend-AID for CICS can also manage region dumps (SYSMDUMP) in the shared directory.

Source Listing Databases and Shared Directories

A source listing shared directory is a specialized VSAM file used by CSS to manage a POOL of source listing databases and to provide LONGNAME support. All products that have source support can use source listing databases. Beginning with Abend-AID Release 9.4, CSS Release 7.9, and Abend-AID for CICS Release 4.5, the Compuware Viewing Facility will allow you to specify a source listing shared directory name. The source listing shared directory name is entered on the CW/VF ENTRY panel and can be used in conjunction with other source listing shared directories and/or source listing files.

Source listing shared directories must be used with XPEDITER/TSO for C language, VisualAge PL/I, and Enterprise PL/I processing if you are using LONGNAME compiler support. Source listing shared directories allow member names greater than the previous DDIO limit of eight characters. The source listing database contains the source listing from the language processor, and the source listing shared directory contains all of the keys necessary to handle LONGNAME support. This allows C language, VisualAge PL/I, and Enterprise PL/I member names to be up to 1024 characters long and may also be mixed case. Multiple source listing databases can be attached to a single source listing shared directory.

For C language, VisualAge PL/I, and Enterprise PL/I LONGNAME compiler support, the Compuware language preprocessor for the programming language being used accepts the source code, invokes the specified compiler, and waits until the compilation step is complete. It then analyzes the output, produces the source listing member, and writes the source listing member to the source listing database and special source listing keys to the source listing shared directory.

When pre- or postprocessing programs that use LONGNAMEs, your JCL should point to the source listing shared directory. When pre- or postprocessing non-LONGNAME programs, your JCL should point to a source listing database (or a source listing file if not using source listing shared directories).

CSS Utilities

CSS Utilities walks you step-by-step through CSS Language Processor (LP) functions and DDIO file manipulation using easy-to-use panels. Supported LP functions include generating preprocessing and postprocessing JCL. DDIO file functions include file creation and formatting, file deletion, and member manipulation (for example, locking/unlocking members, copying members, and importing/exporting members).

CSS Utilities may be used with the following products:

- Abend-AID
- Abend-AID for CICS (for source listing and DSECT files only)
- XPEDITER/CICS
- XPEDITER/TSO (for XPEDITER/TSO 7.0 or later C language support, see “Source Listing Databases and Shared Directories”)

For more information on CSS Utilities, refer to **Chapter 3, “CSS Utilities”**.

Batch Utilities

To create and maintain Abend-AID for CICS shared directory and transaction databases, or as an alternative to CSS utilities, four batch utilities are available:

CWDDSUTL

CWDDSUTL is the batch file equivalent to the CSS Utilities DDIO file functions. CWDDSUTL performs all of the same functions as the CSS Utilities. The following products use CWDDSUTL:

- Abend-AID
- Abend-AID for CICS (for source listing and DSECT files only)
- XPEDITER/CICS
- XPEDITER/TSO (for C language, VisualAge PL/I, and Enterprise PL/I LONGNAME support, see "Source Listing Databases and Shared Directories")

CWFXSDUT

CWFXSDUT creates and maintains the transaction databases and shared directories used by Abend-AID for CICS. CWFXSDUT enables you to store and retrieve transaction reports. It also maintains the shared directory that manages transaction databases. (Shared directories are a special type of DDIO file that must be defined as VSAM datasets.)

CWAASDUT

CWAASDUT creates and maintains the files used by an Abend-AID shared directory and its associated abend report files. The shared directory is used to process all directory requests related to an abend report file.

CWDDLPUT

Creates and maintains the Source Listing Databases and Shared Directories used for source support. It also maintains the shared directory that manages Source Listing Databases and long-named members of Source Listing Databases.

Note: DDIO files formatted using one of the above listed utilities cannot be manipulated using any of the others.

For more information on these utilities, refer to **Chapter 4, "Batch File Utility CWDDSUTL"**, **Chapter 5, "Batch File Utility CWFXSDUT"**, **Chapter 6, "Batch File Utility CWAASDUT"**, or **Chapter 7, "Batch File Utility CWDDLPUT"**.

Source Extraction Utility

CWDDUNLP

Extracts source from a DDIO source listing file. The use of this utility is not intended as a replacement for keeping your original source code and provides support for COBOL and PL/I only. For more information on this utility, refer to **Appendix C, "Using the CWDDUNLP Source Extraction Utility"**.

Viewing DDIO File Information

To view status and other information about DDIO files, you can use the DIRX report. The DIRX report is created using the CWDDSUTL, CWFXSDUT, CWAASDUT, or CWDDLPUT utility command DIRX. In addition to a list of the members on the files, DIRX supplies the following information:

- File utilization information, including the total number of members in the file and how many of them have been allocated

- Format information such as the file access method
- Diagnostic information for Compuware (useful for troubleshooting).

For a sample of the DIRX reports, refer to "Using the DIRX Report" on page 2-17.

Note: CSS Utilities DDIO file functions "I" (information) option can also be used instead of CWDDSUTL.

DDIO Terminology

This section describes the parameters used for allocating and formatting a DDIO file.

AUTODELETE

Determines the action taken when an attempt is made to write a member to a full DDIO file. With this parameter, you can specify how or whether various types of members in the DDIO file are automatically deleted when the file becomes full. For more information on using the autodelete process and how CSS defines the oldest member, refer to "Locking Your DDIO Members" on page 2-7.

BLOCKSIZE (BLKSIZE)

Specifies a blocksize for a sequential DDIO file. The default is 4096 (4 KB) and the maximum blocksize allowed is 32767. For a VSAM DDIO file, the blocksize is always equal to the VSAM Control Interval size (CISIZE) minus 7 and cannot be overridden.

EXTENTS

Specifies the maximum number of segments of file storage allocated for a DDIO member. Extents can also be thought of as the maximum number of times CSS retrieves additional space for a member. DDIO extents are similar to the extents that are allocated for MVS datasets. MVS (non-VSAM) datasets are allowed to have up to 16 extents for primary plus secondary allocation. Extents for a DDIO member can be allocated up to the number specified in the EXTENTS parameters of the FORMAT statement. The default for the EXTENTS parameter is 64 and the maximum is 460.

GROUPCOUNT

Specifies how many blocks or VSAM Control Intervals are contained in a DDIO file extent. GROUPCOUNT is similar to the SPACE allocation for an MVS dataset.

Example

For example, an MVS dataset defined with the parameter:

```
SPACE=(4096,(12,12))
```

has a primary and secondary allocation of 12 4096-byte blocks (1 track) for each extent.

For a DDIO file, a GROUPCOUNT of 12 and a BLKSIZE or CISIZE of 4096 would allocate 12 4096-byte blocks for each extent allocated to a DDIO member. A DDIO member extent, however, is not required to end on a track boundary. You can define it to use only part of a track, a full track, or even several tracks depending on the values for GROUPCOUNT and BLKSIZE or CISIZE.

INDEX

Enables indexes to occupy more than one block. This parameter is only applicable for Abend-AID report files.

REPTCOUNT

Specifies the maximum number of members that can exist simultaneously in a DDIO file provided there is enough space. It is similar to the number of directory blocks specified for an MVS PDS.

If the AUTODELETE parameter was specified as YES, DUPS, or STAGED when the file was formatted, you may still be able to add additional members to the DDIO file if there are any members eligible for autodeletion.

The maximum REPTCOUNT for a source listing file formatted with TYPE=SOURCE is 32767.

Selecting a File Access Method

You can allocate DDIO files as either VSAM or physical sequential datasets. No external functional differences exist in the way these two types of datasets interact with Compuware products. The method you choose should be based on your operating system and can be done on a file-by-file basis.

VSAM and sequential DDIO files are processed identically by CSS, and entries can be transferred between files defined with either access method. However, there are advantages and disadvantages to both methods. Since VSAM allocates its buffers above the 16 MB line, CICS users may experience a performance improvement when using VSAM DDIO files. Sequential dataset buffers are allocated below the 16 MB line so applications that have many DDIO files open may experience shortages of below-the-line storage.

Disk space and memory usage are also considerations when choosing a file access method. You can define a sequential DDIO file to use disk space more efficiently than a VSAM file. VSAM DDIO files are limited by the multiple of 2K Control Interval Size (CISIZE) restriction. However, VSAM DDIO files utilize CICS memory much better than sequential DDIO files in the CICS environment. (This factor is not as critical in a batch environment.)

Note: XPEDITER/CICS requires that all source listing DDIO files in a CICS region use the same access method — either all VSAM or all sequential.

VSAM DDIO files can be copied using the IDCAMS REPRO command but none of the file attributes, such as CISIZE, can be changed.

See Table 2-2 for a list of the job names, and the parameters used with those jobs, when allocating or formatting DDIO files.

Table 2-2. Allocating and Formatting DDIO Files

File	Allocate/Format	Type	JCL Name	Parameter Name
Source	Allocate	VSAM	CXALLVS	CXALDDSL
		Sequential	CXALLDS	
	Format	VSAM	CXFMTDS	CXFMTLST
		Sequential	CXFMTDS	CXFMTLST
Abend-AID Report	Allocate	VSAM	CXALLVS	CXALDDAA
		Sequential	CXALLDS	
	Format	VSAM	CXFMTDS	CXAARPT
		Sequential	CXFMTDS	CXAARPT
Abend-AID for CICS Shared Directory	Allocate and Format	VSAM	CXALLMC	

Table 2-2. Allocating and Formatting DDIO Files

File	Allocate/Format	Type	JCL Name	Parameter Name
Abend-AID for CICS Transaction Database	Allocate and Format	VSAM	CXALLVSD	
		Sequential	CXALLBSD	
Source Listing Shared Directory	Allocate and Format	VSAM	CXALLDLP	
Source Listing Database	Allocate and Format	VSAM	CXALLVLP	
		Sequential	CXALLBLP	
Abend-AID Shared Directory	Allocate and Format	VSAM	CXALLDAA	
Abend-AID Report Database	Allocate and Format	VSAM	CXALLVAA	
		Sequential	CXALLBAA	

DDIO File Considerations

To use DDIO files, you must first allocate a file and then format it. Compuware recommends that you use the default allocation and formatting parameters provided in the SLCXCNTL sample library members. Use the initial DDIO file for about a week to get a representative sampling of DDIO member sizes. The DIRX report will display how much space the DDIO members are using. You can use this information to calculate the allocation specifications for your permanent files. Refer to "Allocating a DDIO File" on page 2-9 for more information on allocating your files.

You should consider the following items before you allocate and format a DDIO file:

- How many listings do you plan to put into your DDIO file?
- What is the average size of the listings you plan to keep in this DDIO file?
- Do you plan to use VSAM or sequential DDIO files (or a mixture of both)?
- Will any of the members be automatically deleted when the file becomes full? For more information, see "Locking Your DDIO Members" below.

Sharing Source Listing Files

Source listing files can be shared among all of the Compuware products that use them and also across multiple address spaces or CICS regions. For example, the same source listing file can service two CICS development regions using both Abend-AID for CICS and XPEDITER/CICS and three production regions using only Abend-AID for CICS. It can also provide source support for Abend-AID reports.

Note: XPEDITER/CICS (prior to release 7.6) and XPEDITER/TSO (prior to release 7.3) do not support shared source directories.

Locking Your DDIO Members

The AUTODELETE parameter of the CWDDSUTL, CWFXSDUT, CWAASDUT, or CWDDLPUT utilities determines the action taken when an attempt is made to write a member to a DDIO file that is full. A DDIO file is considered full when all of the groups (EXTENTs) in the file have been allocated or the number of members specified by REPTCOUNT has been reached.

Manual and Automatic Locking

The autodelete process is based on *manual* and *automatic* locks. Manual locks are created online using the **L** line command from any directory screen or with the batch utility

LOCK command. Automatic locks are created whenever a member is added to a DDIO file formatted using the AUTODELETE=DUPS or AUTODELETE=STAGED option. The most recent of all members with the same name is automatically locked. For shared directory usage, this is done at the database level for the selected database.

Members that have been manually locked retain their locks during the autodeletion process. They may be unlocked by using the **U** line command online or with the UNLOCK batch command. A DELETE command in batch, or the **D** line command online, will unlock a member prior to executing the delete.

For source listings, the oldest member is the program with the earliest compile date and time. For diagnostic reports and transaction reports, the oldest member is the report with the earliest creation date and time. For shared directory usage, this is reflected at the database level.

AUTODELETE Options

Since not all of the batch utilities use AUTODELETE in exactly the same way, refer to the appropriate utility chapter for detailed information about AUTODELETE.

- CWDDSUTL utility — see the AUTODELETE parameter under “Command Parameters for DDIO Files” on page 4-6.
- CWFXSDUT utility — see the AUTODELETE parameter under “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.
- CWAASDUT utility — see the AUTODELETE parameter under “Command Parameters for Shared Directories and Report Files” on page 6-5.
- CWDDLPUT utility — see the AUTODELETE parameter under “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

DDIO File Full Condition

A DDIO file full condition exists when new members cannot be added to a full DDIO file, in which case an error is returned. A DDIO file full condition will occur with the following AUTODELETE options:

YES	All members are manually or automatically locked or in use.
DUPS	All members are manually or automatically locked or in use.
STAGED	All members are manually locked or in use.
NO	The file is full.

Resolving a DDIO File Full Condition

- You can perform the following steps to resolve a file full condition:
- Manually delete members from the file.
- Manually unlock members when AUTODELETE=YES, DUPS, or STAGED.
- Allocate and format a larger file, and use the MOVE or COPY commands to transfer the contents of the full file to the new file.
- Use the EXPORT command with the DELETE option to archive all or some members to a sequential file (tape or disk).

  Using EXPORT, space can be freed up without losing members since members can be imported back into the file as needed. This method also avoids the overhead created by formatting new files.
- If you are using the Source Listing Database and Shared Directory DDIO format (required for C Language Processor, VisualAge PL/I, and Enterprise PL/I LONGNAME support), you can set up your shared directory to perform dynamic database creation.

You must first create a DBMODEL in your shared directory to define the parameters that will be used to create and format a new Source Listing Database dynamically.

The DYNCREATE parameter must also be set to YES. The CWPDDIO DD statement must point to your shared directory (NOT to a specific Source Listing Database). When the file-full condition is encountered, a new Source Listing Database will dynamically be created and formatted and the source member created in that database. An informational message is generated to let you know this has occurred.

Allocating a DDIO File

This section contains the formulas for allocating a DDIO file. After you have allocated a temporary file using the sample installation library members, use the following information to calculate the permanent DDIO file. Refer to "Selecting a File Access Method" on page 2-6 before allocating the file.

Step 1. Calculate the average member size in KBs.

You can calculate the average member size in KBs using the DDIO file with the default 4500 4 KB block allocation and the DIRX report. Refer to the DIRX report for the average member size (in groups).

Use the following formula:

```
AVERAGE MEMBER SIZE (IN GROUPS) x GROUPCOUNT x BLOCKSIZE/1024 =
    average member size in KBs
```

Refer to the DIRX report (shown in Figure 2-1) for the necessary information to complete the calculation.

Figure 2-1. DIRX Report for Allocating DDIO Files

```
  DIRX INDD=ABNLFROM

DSN ----------------------------- CX.QA.DDIO.SC.ADNO
VOLSER -------------------------- PRD903
AM(ACCESS METHOD) --------------- VSAM
BLKSIZE ------------------------- 4089
TYPE ---------------------------- SOURCE
REPTCOUNT ----------------------- 200
GROUPCOUNT ---------------------- 8

 ....

MEMBER SUMMARY OF CX.QA.DDIO.SC.ADNO
NO OF MEMBERS ------------------------------- 12
NO OF LOCKED MEMBERS ------------------------- 2
NO OF UNLOCKED MEMBERS ----------------------- 10
NO OF INCOMPLETE MEMBERS --------------------- 0
LARGEST MEMBER SIZE (IN GROUPS) -------------- 5
SMALLEST MEMBER SIZE (IN GROUPS) ------------- 4
AVERAGE MEMBER SIZE (IN GROUPS) -------------- 4.5

SPACE SUMMARY BASED ON AUTODELETE=STAGED
MAXIMUM ALLOCATION GROUPS -------------------- 896
AVAILABLE ALLOCATION GROUPS ------------------ 892
MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 200
AVAILABLE NO OF DIRECTORY ENTRIES ------------ 188

ENTER UTILITY COMMAND:
*** ALL FUNCTIONS SUCCESSFULLY COMPLETED.
```

Note: The DIRX report for Abend-AID for CICS does not show the member size for specific members. You can calculate the size of a specific member by multiplying the number of EXTENTS by the GROUPCOUNT. Then multiply that number by the BLOCKSIZE to determine the member size in bytes.

Step 2. Calculate the DDIO file size.

After determining the average member size in KBs, use the following formula to determine the number of bytes required for the entire DDIO file.

```
average member size in KBs x estimated number of file members =
    DDIO file size in KBs
```

Step 3. Calculate the allocation in blocks.

Next, determine the number of blocks the DDIO file will use by dividing the total number of bytes required for the file by the block size.

```
file size (determined in Step 2) x 1024/BLOCKSIZE = number of blocks
```

Formatting a DDIO file

There are several factors that control whether a file will successfully format. These factors are interdependent — changing the value of one will affect each of the other factors used to format the file. The factors to consider when formatting a DDIO file are:

- Total number of blocks or CIs in the file

- GROUPCOUNT
- REPTCOUNT
- BLKSIZE or CISIZE

Note: The sequential or VSAM file must be allocated with the correct DCB or DEFINE CLUSTER parameters. The BLKSIZE or CISIZE parameter overrides any value specified in the DD statement DCB parameter when the file was allocated. Refer to SLCXCNTL installation library members CXALLDS and CXALLVS for sample values for these parameters.

Formatting Formulas

Use the following formulas to format a DDIO file:

Step 1. Calculate the total number of blocks in the file.

The total number of blocks in the file is calculated in one of two ways depending on the access method.

If you are formatting a sequential DDIO:

```
no. of blocks per track x total no. of tracks in the file =
total no. of blocks
```

If you are formatting a VSAM file, the number of blocks in the file is the total number of CIs allocated in the CLUSTER. You can determine this by calculating:

```
CIs per track x number of tracks = total number of blocks
```

Step 2. Calculate the value for the GROUPCOUNT.

The minimum value for GROUPCOUNT is calculated as:

```
total number of blocks/32767 = GROUPCOUNT
```

You must round this number up to the next whole number.

Step 3. Calculate the minimum value for BLKSIZE or CISIZE.

The minimum value for the BLKSIZE for sequential files, or the CISIZE for VSAM files, is determined by:

```
68 + (2 x REPTCOUNT) + (total no. of blocks/(8 x GROUPCOUNT)) =
     min. BLKSIZE or CISIZE
```

To determine the minimum blocksize for a transaction database, Abend-AID report file, Source Listing Database, or shared directory, add 100 bytes to the final calculation.

The REPTCOUNT and GROUPCOUNT are the values specified when the file is formatted, and BLOCKS is the total number of physical blocks (CIs) in the file.

If you are formatting a sequential file, you should round up to the next best blocking factor for your DASD device type.

If you are formatting a VSAM file, round up the value from the formula to the next 2K multiple since VSAM files are allocated in multiples of 2K. The number of VSAM

CIs/track depends on the CISIZE and the type of DASD being used. Refer to Table 2-3 for the amount of VSAM space allocated per track by CISIZE.

Table 2-3. Amount of VSAM Space Allocated per Track by CISIZE

CISIZE	Space Allocated per Track	
	3380 DASD	3390 DASD
4K	40K	48K
6K	42K[1]	48K
8K	40K	48K
10K	40K	50K
12K	36K	48K
14K	42K[1]	49K
16K	32K	48K
18K	36K	54K
20K	40K	50K
22K	44K[1]	49.5K
24K	44K	48K
26K		52K
28K		49K
30K		50K
32K		48K

[1] For 3380 DASD, these CISIZEs provide good space usage.

Understanding DDIO File Limitations

There are some limitations on the physical structure of the DDIO file that may affect how you format the DDIO file. These limitations include the following:

- The number of allocation groups cannot exceed 32767. Allocation groups refers to the number of groups available in which a report can be stored. If you exceed 32767 allocation groups, you will receive the message:

  ```
  NUMBER OF ALLOCATION GROUPS MUST NOT EXCEED 32767
  ```

  You can avoid this condition by ensuring that the GROUPCOUNT value is large enough.

- The BLKSIZE or CISIZE must be large enough to contain all the pointers to the directory entries defined for the file. The number of directory entries is set by the REPTCOUNT parameter. The size of each directory entry varies based on the EXTENTS and GROUPCOUNT parameters. If you need to increase the BLKSIZE, you will receive the following message:

  ```
  BLOCKSIZE TOO SMALL FOR HEADER RECORD
  ```

You can avoid this condition by ensuring that the BLOCKSIZE or CISIZE value is large enough.

Allocating and Formatting Example

This section shows an example of allocating and formatting a source listing file but can be used for any type of DDIO file. A source listing file was previously allocated based on the SLCXCNTL installation library member CXFMTLST. We have stored listings in it for a week to determine how much space the files are using.

Allocating the VSAM Cluster

This DDIO file will be used with XPEDITER/CICS, so the access method we will choose is VSAM (on a 3390 DASD). Use the following steps to allocate the VSAM cluster.

Step 1. Determine the average DDIO file member size.

To determine the average size of each listing, run a DIRX report on the sample temporary file (the file used with the sample JCL from the installation library). For this example, the average size of each listing is 240K.

Step 2. Calculate the DDIO file size.

In this example, the file size was determined by assuming that there would be 1000 source listing members.

```
1000 members x 240K average listing size = 240000K
```

Step 3. Calculate the allocation in blocks.

For this example, a CISIZE of 6K was assumed.

```
240000K/48K = 5000 number of tracks required
```

Next, determine the number of cylinders to allocate.

```
5000/15 = 334 cylinders to allocate
```

One more cylinder will be added to make it an even multiple of 5 and to allow some space for the DDIO file directory.

Formatting the File

We can now format the DDIO file, using the following formulas.

Step 1. Calculate the total number of blocks in the file.

Determine the number of tracks:

```
335 cylinders x 15 tracks per cylinder = 5025 tracks
```

Next, determine the total number of blocks in the file:

```
8 CIs per track x 5025 tracks = 40200 total blocks in the file
```

Step 2. Calculate the value for the GROUPCOUNT.

```
40200/32767 = 1.23 (round up) minimum GROUPCOUNT is 2
```

Step 3. Calculate the minimum value for CISIZE.

```
68 + (2 x 1000) + (40200/(8 x 2)) = 4580.5 minimum BLKSIZE/CISIZE
```

Formatting Specifications

Format the file with the following FORMAT parameters:

```
FORMAT TYPE=SOURCE,AD=STAGED,EXTENTS=460,RC=1000,GC=2
```

The parameters used in the example specify the following:

AD=STAGED

Because this DDIO file will store development source listings, we used AUTODELETE=STAGED. When a new member is added to the DDIO file, all automatically locked members with the same name are unlocked — except the member with the most recent compile date and time. If the file is full, the oldest unlocked member not in use is chosen for autodeletion.

Note: For more information on locking DDIO members, see "Locking Your DDIO Members" on page 2-7. For more information on the AUTODELETE parameter used by Compuware utilities, refer to:

- CWDDSUTL utility — "Command Parameters for DDIO Files" on page 4-6,
- CWFXSDUT utility — "Command Parameters for Shared Directories and Transaction Databases" on page 5-5,
- CWAASDUT utility — "Command Parameters for Shared Directories and Report Files" on page 6-5,
- CWDDLPUT utility — "Command Parameters for Shared Directories and Source Listing Databases" on page 7-6.

EXTENTS=460

The smallest member (1 EXTENT) will use 12K bytes:

```
1 x 2 x 6K bytes = 12K bytes
```

The largest member that can be stored is 5520K bytes:

```
460 x 2 x 6K = 5520K bytes
```

There are *approximately* 20000 available 2-block EXTENTS on the file.

This number is calculated using the following formula (round calculation down):

```
total # of blocks in file - (# of directory blocks + 2)/GROUPCOUNT
```

Note: You won't know the exact number of directory blocks until the file is formatted and you run a DIRX report. You can also read the exact number of allocated groups directly from the DIRX report.

RC=1000

This file can contain 1000 unique programs (REPTCOUNT) that use an average of 20 EXTENTS each.

```
20000 EXTENTS/1000 programs = 20 EXTENTS = 240K bytes
```

This file can store 43 programs that are 5520K bytes long (460 x 2 x 6K) with 220 EXTENTS remaining:

```
20000/460 = 43.5 (round down) and 20000 - (460 x 43) = 220 remaining
```

Note: This file can store any intermediate combination of the above.

Tuning Your DDIO File

How you format your file depends on how you plan to use it. You can use the same file allocation with different combinations of FORMAT and BLKSIZE/CISIZE parameters and come up with different capacities. Performing these calculations, using several different combinations of GROUPCOUNT and BLKSIZE/CISIZE, will help you determine the best values to use for formatting a DDIO file that meets your needs.

The Compuware DDIO Calculator is a tool that can be used to calculate your allocation and formatting of DDIO files (source and report) and databases (source, report and transaction). Although an existing file is not required to use this tool, it is highly recommended for use as a model to fine tune. Otherwise, calculating file size and determining optimal formatting parameters will result in trial and error.

Note: It is not used for shared directory allocation or tuning.

The DDIO Calculator is available as a menu selection on the CSS Utilities main menu (see Figure 3-1 on page 3-2); as a panel option on the CSS Utilities Source Listing File Facility screen (Figure 3-6 on page 3-11) and the Abend-AID Report File Facility screen (Figure 3-42 on page 3-37); or, optionally, as a TSO REXX command “DDIOCALC”, found in the CSS-supplied SLCXEXEC library. The DDIO Calculator usage is described further in “Compuware DDIO Calculator — DDIOCALC” on page 3-61.

Tuning the Sample File

If the largest program that will ever be compiled will fit into a 5520K byte DDIO file member, the values in the preceding example will work fine. If you have larger programs, you need to increase the GROUPCOUNT. Suppose GROUPCOUNT=3 is used with CISIZE=6K. Calculate the following values.

- The minimum BLKSIZE/CISIZE is 3743.

  ```
  68 + (2 x 1000) + (40200/(8 x 3)) = 3743
  ```

- The smallest member uses 18K bytes (1 x 3 x 6K). The largest member that can be stored is 8280K bytes (460 x 3 x 6K).
- There are approximately 13350 available 3-block EXTENTS on the file.
- This file can contain 1000 unique programs with an average of 13.3 EXTENTS (239.4K)
- This file can store 29 programs that are 8280K bytes long with 10 extents remaining.

Decreasing CISIZE

Let us see if this sample DDIO file can be decreased to a 4K size with a GROUPCOUNT of 4:

```
68 + (2 x 1000) + (60300 / (8 x 4)) = 3952.38
```

So, if the following parameters are used:

```
FORMAT TYPE=SOURCE,RC=1000,GC=4,EXTENTS=460,AD=STAGED
```

the file has the following characteristics:

- The smallest member uses 16K bytes (1 x 4 x 4K bytes); the largest member that can be stored is 7360K bytes (460 x 4 x 4K).
- There are approximately 15000 available 4-block EXTENTS on the file.
- This file can contain 1000 unique programs with an average of 15 EXTENTS (240K).

- This file can store 32 programs that are 7360K bytes long with 280 EXTENTS remaining.

Refer to Table 2-4 for a comparison of the values used to format a file.

Table 2-4. DDIO File Size Comparison

Format Values	Total Blocks in File	Smallest/ Largest Member	Available Extents on File	Number of Programs/ Average Extents	Min BLKSIZE/CISIZE
GC=2, CISIZE=6K	40200	12K/5520K	20000 2-block	1000/20 extents	4580.5
GC=3, CISIZE=6K	40200	18K/8280K	13350 3-block	1000/13.3 extents	3743
GC=4, CISIZE=4K	60300	16K/7360K	15000 4-block	1000/15 extents	3952.38

You may find that even though you have allocated enough space to hold all of the members you want for a specified time period, they are being deleted too soon. Print a DIRX report to determine if all of the space you have allocated is used. If not, you may have reached the limit specified by the REPTCOUNT parameter when you formatted your file. When this limit is reached, the autodelete process will be initiated when adding the next member — regardless of how full the file is. If the file is not full, check the REPTCOUNT in the DIRX and the total number of members in the file.

Reducing the Size of DDIO Source Listing Members

Language processor options exist to reduce the amount of DASD used by each source listing. You can use the following language processor commands and parameters to minimize space usage:

```
DDIO(OUTPUT(NOXREF,NOLIST,NODMAP,NOFIND,COMPRESS))
COBOL(OUTPUT(PRINT,DDIO))
PROCESSOR(OUTPUT(NOPRINT,NODDIO))
```

Note: For XPEDITER/TSO users, the FIND command is recommended instead of the NOFIND command. XPEDITER/TSO requires DDIO(OUTPUT(FIND)) in order to use the enhanced FIND feature.

When these parameters are specified, the Cross Reference and LIST (or PMAP) information from the listing is not stored in the DDIO file. The product's performance at execution time is not compromised.

The DDIO parameter offers significant space reduction potential. Including the NOXREF and NOLIST options with this parameter can reduce DASD usage by 30 - 60 percent.

Using the DIRX Report

This section provides examples of the DIRX report format with descriptions of each of the fields on the report. The following samples are provided:

- Source Listing File (Figure 2-2 on page 2-18)
- Abend-AID Report File (Figure 2-4 on page 2-25)
- Transaction Database (Figure 2-6 on page 2-31)
- Shared Directory Report (Figure 2-8 on page 2-37)
- Source Listing Database Report (Figure 2-18 on page 2-50)
- Source Listing Shared Directory Report (Figure 2-20 on page 2-57)

Using the Previous DIRX Report Format

Beginning in CSS 4.0, the DIRX report layout changed significantly for easier viewing. However, you can still use the previous format with a simple rename procedure.

To use the old DIRX report format, rename CWDDUDIR to another name. Then rename CWDDUDRO to CWDDUDIR.

Source Listing File DIRX Report

A sample DIRX report for a source listing file is shown in Figure 2-2 on page 2-18.

Figure 2-2. Sample DIRX Report for a Source Listing File (*Part 1 of 2*)

```
ENTER UTILITY COMMAND:
  DIRX INDD=ABNLFROM

DSN ---------------------------- CX.QA.DDIO.SC.ADNO
VOLSER -------------------------- PRD903
AM(ACCESS METHOD) --------------- VSAM
BLKSIZE ------------------------- 4089
TYPE ---------------------------- SOURCE
REPTCOUNT ----------------------- 200
GROUPCOUNT ---------------------- 8
AUTODELETE ---------------------- STAGED
INDEX --------------------------- NON-EXT
EXTENTS ------------------------- 200
OTHER INFORMATION
  BLOCKS IN FILE ---------------- 7200
  ALLOCATION GROUPS ------------- 896
  USED ALLOCATION GROUPS -------- 54(6%)
  FILE FORMATTED WITH RELEASE --- 6.0.0
  REQUIRED MINIMUM RELEASE ------ 5.1.0
  CURRENT REPORT NUMBER --------- 263
  DIRECTORY ENTRIES/BLOCK ------- 8
  DIRECTORY BLOCKS -------------- 25
  FILE FORMAT ------------------- EXTENDED
  DDIO EXTENDED DIR SPACE ------- 6
  USER EXTENDED DIR SPACE ------- 0

PROGRAM    LIST NUMBER   COMP DATE  TIME    RC  LANGUAGE ---BYTES---             GROUPS
FEA02A1    (L)     184  20 OCT 2001 14.06   00  HLASM       159K                   50    51    52    53    54
                                                                                                                TOTAL GROUPS....5
FEA02A1            183  20 OCT 2001 14.05   00  HLASM       159K                   45    46    47    48    49
                                                                                                                TOTAL GROUPS....5
FEA02A1            182  20 OCT 2001 14.05   00  ASMH        127K                   41    42    43    44
                                                                                                                TOTAL GROUPS....4
FEA02A1    (M)     181  20 OCT 2001 14.04   00  ASMH        127K                   37    38    39    40
                                                                                                                TOTAL GROUPS....4
FEA02A1            180  20 OCT 2001 14.03   00  ASMF        127K                   33    34    35    36
                                                                                                                TOTAL GROUPS....4
FEA02A1            179  20 OCT 2001 14.02   00  ASMF        127K                   29    30    31    32
                                                                                                                TOTAL GROUPS....4
FEA02A1            178  20 OCT 2001  14.00  00  HLASM       159K                   24    25    26    27    28
                                                                                                                TOTAL GROUPS....5
FEA02A1            177  20 OCT 2001  14.00  00  HLASM       159K                   19    20    21    22    23
                                                                                                                TOTAL GROUPS....5
FEA02A1            176  20 OCT 2001  13.58  00  ASMH        159K                   14    15    16    17    18
                                                                                                                TOTAL GROUPS....5
FEA02A1            175  20 OCT 2001  13.57  00  ASMH        159K                    9    10    11    12    13
                                                                                                                TOTAL GROUPS....5
FEA02A1            174  20 OCT 2001  13.54  00  ASMF        127K                    5     6     7     8
                                                                                                                TOTAL GROUPS....4
FEA02A1            173  20 OCT 2001  13.53  00  ASMF        127K                    1     2     3     4
```

Figure 2-3. Sample DIRX Report for a Source Listing File (*Part 2 of 2*)

```
MEMBER SUMMARY OF CX.QA.DDIO.SC.ADNO

 NO OF MEMBERS ------------------------------- 12
 NO OF LOCKED MEMBERS ------------------------ 2
 NO OF UNLOCKED MEMBERS ----------------------- 10
 NO OF INCOMPLETE MEMBERS --------------------- 0
 LARGEST MEMBER SIZE (IN GROUPS) -------------- 5
 SMALLEST MEMBER SIZE (IN GROUPS) ------------- 4
 AVERAGE MEMBER SIZE (IN GROUPS) -------------- 4.5

 SPACE SUMMARY BASED ON AUTODELETE=STAGED
 MAXIMUM ALLOCATION GROUPS -------------------- 896
 AVAILABLE ALLOCATION GROUPS ------------------ 892
 MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 200
 AVAILABLE NO OF DIRECTORY ENTRIES ------------ 188

 ENTER UTILITY COMMAND:
 *** ALL FUNCTIONS SUCCESSFULLY COMPLETED.
```

Source Listing File DIRX Report Fields

The following is a description of the Source Listing File DIRX Report fields.

ENTER UTILITY COMMAND: CWDDSUTL prompt.

DIRX INDD=ABNLFROM: The CWDDSUTL command used for this job.

DSN: The dataset name of the source listing file.

VOLSER: The volume serial number of the source listing file.

AM(ACCESS METHOD): This file's access method — either VSAM or BSAM (sequential).

BLKSIZE: For sequential files, this is the value of the FORMAT command's BLKSIZE parameter that was set when the file was formatted. For VSAM files, this is the value specified on the RECORDSIZE parameter when the cluster was defined. This value must be seven bytes less than the CISIZE specified when the cluster was defined.

TYPE: Indicates the type of DDIO file. In this case the file is a source listing file.

REPTCOUNT: The maximum number of members (source listings) that the source listing file can contain. This value was specified when the file was formatted.

GROUPCOUNT: The number of blocks of data that can be contained in each group (extent). This value was specified when the file was formatted.

AUTODELETE: The AUTODELETE option that was specified when the file was formatted.

INDEX: Indicates whether the file was formatted to allow indexes to occupy more than one block. This field is useful only for Abend-AID report files. NON-EXT or EXT may appear in this field.

EXTENTS: The largest number of groups (extents) that can be used by a single member.

Other Information

This section of the report lists additional characteristics about the file.

BLOCKS IN FILE: The number of blocks allocated for the file.

ALLOCATION GROUPS: The number of groups available in which a report can be stored. This number is derived by subtracting the number of blocks required by the directory and header record from the total number of blocks in the file and then dividing the result by the GROUPCOUNT value.

USED ALLOCATION GROUPS: The number and percentage of allocation groups that contain data.

FILE FORMATTED WITH RELEASE: The release level of CWDDSUTL that was used to format the file.

REQUIRED MINIMUM RELEASE: The minimum release level of CWDDSUTL that can be used to manage the file.

CURRENT REPORT NUMBER: The number that will be assigned to the next listing written to the file. Report numbers are assigned sequentially.

DIRECTORY ENTRIES/BLOCK: The number of directory entries that will fit per block on the file. This number is dependent on the EXTENTS and BLKSIZE values. As the number of EXTENTS increases, the number of directory entries per block decreases (provided the BLKSIZE does not change).

DIRECTORY BLOCKS: A value derived by dividing the REPTCOUNT used to format the file by the number of directory entries per block.

FILE FORMAT: Indicates whether additional bytes are in the directory.

DDIO EXTENDED DIR SPACE: A value used by Compuware for problem determination.

USER EXTENDED DIR SPACE: A value used by Compuware for problem determination.

Column Headings

The header line for the body of the DIRX report includes the following columns:

PROGRAM: The name of the program. If followed by (L), the member is automatically locked. If followed by (M), the member is manually locked.

LIST NUMBER: The number assigned to this member by CWDDSUTL.

COMP DATE: The date the program was compiled.

TIME: The time of day the program was compiled.

RC: The return code from the step that created the member.

LANGUAGE: The program language compiler and version. The following codes can be found in the LANGUAGE column:

ASMF	Assembler F
ASMH	Assembler H
HLASM	High Level Assembler 1.1
HLASM R2	High Level Assembler 1.2
HLASM R3	High Level Assembler 1.3
HLASM R4	High Level Assembler 1.4
OSVSCOB	OS/VS COBOL
OSVSCOBOPT	OS/VS COBOL/CA-Optimizer
C/ESA V3R2	“C” for ESA V3R2
C/390 V1R2	OS/390 “C” V1R2
C/390 V1R3	OS/390 “C” V1R3
C/390 V2R4	OS/390 “C” V2R4
C/390 V2R5	OS/390 “C” V2R5
C/390 V2R6	OS/390 “C” V2R6
C/390 V2R9	OS/390 “C” V2R9
C/390 V2RA	OS/390 “C” V2R10
COBIIR2	VS COBOL II Release 2
COBIIR2OPT	VS COBOL II Release 2/CA-Optimizer II
COBIIR3	VS COBOL II Release 3
COBIIR3OPT	VS COBOL II Release 3/CA-Optimizer II
COBIIR4	VS COBOL II Release 4
COBIIR4 JA	VS COBOL II Release 4 with Japanese Language Support
COBIIR4OPT	VS COBOL II Release 4/CA-Optimizer II
COBII JA	VS COBOL II with Japanese Language Support
COBOL/370	IBM SAA AD/Cycle COBOL/370
COBOL/370 JA	IBM SAA AD/Cycle COBOL/370 with Japanese Language Support
COBOL/370OPT	IBM SAA AD/Cycle COBOL/370/CA-Optimizer II
COBOL/390	COBOL for OS/390 and VM

COBOL/390M	COBOL for OS/390 and VM with MLE (Millennium Language Extensions)
COBOL/MVSM	COBOL for MVS and VM with MLE
COB/370JA	COBOL/370 Japanese Language Support
COB/390OPM	COBOL/390 MLE CA-Optimizer
COB/390OPT	COBOL/390 CA-Optimizer
COB/MVS	COBOL for MVS and VM
COB/MVS JA	COBOL for MVS and VM with Japanese Language Support
COB/MVSOPM	COBOL/MVS MLE CA-Optimizer
COB/MVSOPT	COBOL/MVS CA-Optimizer
COBOL Z/OS	Enterprise COBOL for z/OS and OS/390.
COBOL Z/OSJA	Enterprise COBOL for z/OS and OS/390 with Japanese language support.
COB/Z V3R2	Enterprise COBOL for z/OS and OS/390 Release 3.2
COB Z/OSJA	Enterprise COBOL for z/OS and OS/390 Release 3.2 with Japanese Language Support
PL/I V1R5	PL/I V1
PL/I V2	PL/I V2
PL/I AD V1	PL/I for MVS and VM
PL1VA V2R2	PL/I VisualAge V2R2
PL1EN V3R1	Enterprise PL/I for z/OS
PL1EN V3R2	Enterprise PL/I for z/OS

BYTES: The number of bytes this member occupies on the source file.

GROUPS: The allocation group numbers occupied by the source listing. For example, source listing number 183 for program FEA02A1 occupies group numbers 45, 46, 47, 48, and 49.

TOTAL GROUPS: The total number of allocation groups occupied by the source listing.

Member Summary

The following information appears at the end of the report.

NO OF MEMBERS: The total number of members in the source listing file.

NO OF LOCKED MEMBERS: The total number of locked members in the source listing file.

NO OF UNLOCKED MEMBERS: The total number of unlocked members in the source listing file.

NO OF INCOMPLETE MEMBERS: The total number of members in the source listing file that are missing data and therefore cannot be viewed. Incomplete members can result when a file full condition exists while the member is being written to the file or when certain system failures occur.

LARGEST MEMBER SIZE (IN GROUPS): The size of the largest member in the source listing file, expressed in groups.

SMALLEST MEMBER SIZE (IN GROUPS): The size of the smallest member in the source listing file, expressed in groups.

AVERAGE MEMBER SIZE (IN GROUPS): The size of the average member in the source listing file, expressed in groups.

Space Summary Based on AUTODELETE

The following fields describe the characteristics of the source listing file using the specified AUTODELETE option.

MAXIMUM ALLOCATION GROUPS: The number of allocation groups that can exist in the source listing file.

AVAILABLE ALLOCATION GROUPS: The number of allocation groups occupied by unlocked members plus the number of unused allocation groups.

MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT): The maximum number of members (source listings) the source listing file can contain.

AVAILABLE NO OF DIRECTORY ENTRIES: The difference between the maximum number of members the file can contain and the number of locked members (manually locked when AUTODELETE=STAGED).

Abend-AID Report File DIRX Report

A sample Abend-AID Report File DIRX Report is shown in Figure 2-4 on page 2-25. Ellipses indicate omitted portions.

Figure 2-4. Sample DIRX Report for an Abend-AID Report File (*Part 1 of 2*)

```
ENTER UTILITY COMMAND:
  DIRX  INDD=ABNLFROM
DSN ---------------------------- CWV.AA.R80.RPTFILE
VOLSER -------------------------- PRD901
AM(ACCESS METHOD) --------------- VSAM
BLKSIZE ------------------------- 4089
TYPE ---------------------------- Abend-AID
REPTCOUNT ----------------------- 100
GROUPCOUNT ---------------------- 4
AUTODELETE ---------------------- YES
INDEX --------------------------- EXT
EXTENTS ------------------------- 64
OTHER INFORMATION
  BLOCKS IN FILE ---------------- 1800
  ALLOCATION GROUPS ------------- 448
  USED ALLOCATION GROUPS -------- 68(15%)
  FILE FORMATTED WITH RELEASE --- 6.0.0
  REQUIRED MINIMUM RELEASE ------ 5.6.0
  CURRENT REPORT NUMBER --------- 92
  DIRECTORY ENTRIES/BLOCK ------- 18
  DIRECTORY BLOCKS -------------- 6
  FILE FORMAT ------------------- EXTENDED
  DDIO EXTENDED DIR SPACE ------- 6
  USER EXTENDED DIR SPACE ------- 0

JOBNAME    REPT NUMBER JESID     CODE    DATE       TIME OF DAY  DESCRIPTION-BYTES-EXTENTS
JOBNAME1            91 JOB16100  SOC7  28 OCT 2001  11.29.33.9   GRANT 7541    15K      46
                                                                                                TOTAL GROUPS....1

JOBNAME1            90 JOB16054  SOC7  28 OCT 2001  11.25.27.2   GRANT 7541    15K      45
                                                                                                TOTAL GROUPS....1

JOBNAME1            89 JOB16031  SOC7  28 OCT 2001  11.23.33.5   GRANT 7541    15K      44

. . .
                                                                                                TOTAL GROUPS....1

JOBNAME2 (L)        23 JOB08329  SOC7  03 FEB 2001  15.05.53.9   CLM/JCL       15K      28
                                                                                                TOTAL GROUPS....1

JOBNAME2 (L)        22 JOB08327  SOC7  03 FEB 2001  15.05.31.2   CLM/JCL       15K      27
                                                                                                TOTAL GROUPS....1
. . .


JOBNAME3             2 JOB16243  08097 20 JAN 2001   9.23.08.0   JCLPLI2L      31K       3     4
                                                                                                TOTAL GROUPS....2

JOBNAME4             1 JOB21876  SNAP  13 JAN 2001  14.51.43.7   JCLSNAP       31K       1     2
                                                                                                TOTAL GROUPS....2
```

Figure 2-5. Sample DIRX Report for an Abend-AID Report File (*Part 2 of 2*)

```
MEMBER SUMMARY OF CWV.AA.R80.RPTFILE

NO OF MEMBERS -------------------------------- 56
NO OF LOCKED MEMBERS ------------------------- 4
NO OF UNLOCKED MEMBERS ----------------------- 52
NO OF INCOMPLETE MEMBERS --------------------- 0
LARGEST MEMBER SIZE (IN GROUPS) -------------- 6
SMALLEST MEMBER SIZE (IN GROUPS) ------------- 1
AVERAGE MEMBER SIZE (IN GROUPS) -------------- 1.2

SPACE SUMMARY BASED ON AUTODELETE=YES
MAXIMUM ALLOCATION GROUPS -------------------- 448
AVAILABLE ALLOCATION GROUPS ------------------ 444
MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 100
AVAILABLE NO OF DIRECTORY ENTRIES ------------ 96

ENTER UTILITY COMMAND:
*** ALL FUNCTIONS SUCCESSFULLY COMPLETED.
```

Abend-AID Report File DIRX Report Fields

The following is a description of the Abend-AID Report File DIRX Report fields.

ENTER UTILITY COMMAND: CWDDSUTL prompt.

DIRX INDD=ABNLFROM: The CWDDSUTL command used for this job.

DSN: The dataset name of the report file.

VOLSER: The volume serial number of the report file.

AM(ACCESS METHOD): The file's access method— either VSAM or BSAM (sequential).

BLKSIZE: For sequential files, this is the value of the FORMAT command's BLKSIZE parameter that was set when the file was formatted. For VSAM files, this is the value specified on the RECORDSIZE parameter when the cluster was defined. This value must be seven bytes less than the CISIZE specified when the cluster was defined.

TYPE: Indicates the type of DDIO file. In this instance, the report file was formatted for Abend-AID.

REPTCOUNT: The maximum number of members (diagnostic reports) that the report file can contain, this value was specified when the file was formatted.

GROUPCOUNT: The number of blocks of data that can be contained in each group (extent). This value was specified when the file was formatted.

AUTODELETE: The AUTODELETE option that was specified when the file was formatted.

INDEX: Indicates whether the file was formatted to allow indexes to occupy more than one block with INDEX=EXT. This field is useful only for Abend-AID report files. NON-EXT or EXT may appear in this field.

EXTENTS: The largest number of groups (extents) that can be used by a single member.

Other Information

This section of the report lists additional characteristics about the file.

BLOCKS IN FILE: The number of blocks allocated for the file.

ALLOCATION GROUPS: The number of groups available in which a report can be stored. This number is derived by subtracting the number of blocks required by the directory and header record from the total number of blocks in the file and then dividing the result by the GROUPCOUNT value.

USED ALLOCATION GROUPS: The number and percentage of allocation groups that contain data.

FILE FORMATTED WITH RELEASE: The release level of CWDDSUTL that was used to format the file.

REQUIRED MINIMUM RELEASE: The minimum release level of CWDDSUTL that can be used to manage the file.

CURRENT REPORT NUMBER: The number that will be assigned to the next member written to the file. Report numbers are assigned sequentially.

DIRECTORY ENTRIES/BLOCK: The number of directory entries that will fit per block on the file. This number is dependent on the EXTENTS and BLKSIZE values. As the number of EXTENTS increases, the number of directory entries per block decreases (provided the BLKSIZE does not change).

DIRECTORY BLOCKS: A value derived by dividing the REPTCOUNT used to format the file by the number of directory entries per block.

FILE FORMAT: Indicates whether additional bytes are in the directory.

DDIO EXTENDED DIR SPACE: A value used by Compuware for problem determination.

USER EXTENDED DIR SPACE: A value used by Compuware for problem determination.

Column Headings

The header line for the body of the DIRX report includes the following columns:

JOBNAME: The name from the JOB card. If followed by (M), the member is manually locked and will not be automatically deleted when the file becomes full.

REPT NUMBER: The number assigned to this member when it was created.

JESID: The job number assigned by the Job Entry System at run time.

CODE: The abend code that was returned when the job failed.

DATE: The date the job was run.

TIME OF DAY: The time when the job was run.

DESCRIPTION: By default, the first 10 characters of the programmer name field on the JOB card. When using Abend-AID exit CWEXIT02, or when calling SNAP-AID, you can override this value, in which case the new value will be shown. See the Abend-AID documentation for more information.

BYTES: The number of bytes that this report occupies on the file.

EXTENTS: The group numbers that this section of the report occupies.

TOTAL GROUPS: The total number of allocation groups that the diagnostic report occupies.

Member Summary

The following information appears at the end of the report.

NO OF MEMBERS: The total number of members in the report file.

NO OF LOCKED MEMBERS: The total number of locked members in the report file.

NO OF UNLOCKED MEMBERS: The total number of unlocked members in the report file.

NO OF INCOMPLETE MEMBERS: The total number of members in the report file that are missing data and therefore cannot be viewed. Incomplete members can result when a file full condition exists while the member is being written to the file or when certain system failures occur.

LARGEST MEMBER SIZE (IN GROUPS): The size of the largest member in the report file, expressed in groups.

SMALLEST MEMBER SIZE (IN GROUPS): The size of the smallest member in the report file, expressed in groups.

AVERAGE MEMBER SIZE (IN GROUPS): The average member size in the report file, expressed in groups.

Space Summary Based on AUTODELETE

The following fields describe the characteristics of the report file using the specified AUTODELETE option.

MAXIMUM ALLOCATION GROUPS: The number of allocation groups that can exist in the report file.

AVAILABLE ALLOCATION GROUPS: The number of allocation groups occupied by unlocked members plus the number of unused allocation groups.

MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT): The maximum number of members (diagnostic reports) the report file can contain.

AVAILABLE NO OF DIRECTORY ENTRIES: The difference between the maximum number of members the file can contain and the number of locked members.

Transaction Database DIRX Report

A sample DIRX report for a Transaction Database is shown in Figure 2-6 on page 2-31.

Figure 2-6. Sample DIRX Report for a Transaction Database. (*Part 1 of 2*)

```
DIRX FOR DATABASE: SJPTXS0.MP2.DBIO2
           VOLSER: PRD907


DATABASE IS CONNECTED
CATALOG DSN: SJPTXS0.MP2.CAT
     VOLSER: PRD924
DSN ---------------------------- SJPTXS0.MP2.DBIO2
VOLSER -------------------------- PRD907
AM(ACCESS METHOD) --------------- BSAM
BLKSIZE ------------------------- 3584
TYPE ---------------------------- CICS
REPTCOUNT ----------------------- 500
GROUPCOUNT ---------------------- 1
AUTODELETE ---------------------- NO
INDEX --------------------------- NON-EXT
EXTENTS ------------------------- 64
OTHER INFORMATION
  BLOCKS IN FILE ---------------- 2925
  ALLOCATION GROUPS ------------- 2891
  USED ALLOCATION GROUPS -------- 1397(48%)
  FILE FORMATTED BY DDIO RELEASE- 7.0.0
  REQUIRED MINIMUM DDIO RELEASE - 7.0.0
  CURRENT REPORT NUMBER --------- 69
  DIRECTORY ENTRIES/BLOCK ------- 16
  DIRECTORY BLOCKS -------------- 32
  FILE FORMAT ------------------- EXTENDED
  DDIO EXTENDED DIR SPACE ------- 6
  USER EXTENDED DIR SPACE ------- 0


TRAN/TERM  REPT NUMBER  SYSTEM   CODE    DATE       TIME OF DAY  PROGRAM--NODE--EXTENTS
RLM1 A243          709 H01AC184  AEI1  OCT 21 2001  14.15.59.5   RLMCICS1TFHA243    987   988   989   990   991   992   993   994
                                                                                      995   996   997   998   999  1000  1001  1002
                                                                                     1003  1004  1005
                                                                                                          TOTAL GROUPS    19


PYU1 NONE          705 H01AC183  ASCR  OCT 21 2001  14.14.09.9   PAYUPAA1           970   971   972   973   974   975   976   977
                                                                                      978   979   980   981   982   983   984   985
                                                                                      986
                                                                                                          TOTAL GROUPS    17

. . .
. . .
. . .
```

Figure 2-7. Sample DIRX Report for a Transaction Database. (*Part 2 of 2*)

```
. . .
. . .
. . .
SAMP %000          34 H01AC189  AEIO  OCT 21 2001  16.45.29.7 HILL     PLAYBACK    151   152   153   154   155   156   157   158
                                                                                    159   160   161   162   163   164   165   166
                                                                                    167   168   169   170   171   172   173
                                                                                                                TOTAL GROUPS    23

SAMP %001          57 H01AC189  ASRA  OCT 21 2001  16.43.34.8 HILL     PLAYBACK    576   577   578   579   580   581   582   583
                                                                                    584   585   586   587   588   589   590   591
                                                                                    592   593   594   595
                                                                                                                TOTAL GROUPS    20


MEMBER SUMMARY OF SJPTXS0.MP2.DBIO2

NO OF MEMBERS -------------------------------- 68
NO OF LOCKED MEMBERS ------------------------- 12
NO OF UNLOCKED MEMBERS ----------------------- 56
NO OF INCOMPLETE MEMBERS --------------------- 0
LARGEST MEMBER SIZE (IN GROUPS) -------------- 25
SMALLEST MEMBER SIZE (IN GROUPS) ------------- 17
AVERAGE MEMBER SIZE (IN GROUPS) -------------- 20.5

SPACE SUMMARY BASED ON AUTODELETE=NO
MAXIMUM ALLOCATION GROUPS -------------------- 2891
AVAILABLE ALLOCATION GROUPS ------------------ 1494
MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 500
AVAILABLE NO OF DIRECTORY ENTRIES ------------ 432


*  COMMENT
*** ALL FUNCTIONS SUCCESSFULLY COMPLETED
```

Transaction Database DIRX Report Fields

The following is a description of the Transaction Database DIRX Report fields.

DIRX FOR DATABASE: The dataset name of the transaction database.

VOLSER: The volume serial number of the transaction database.

DATABASE IS CONNECTED: Indicates that the transaction database is attached to a shared directory.

CATALOG DSN: The dataset name of the shared directory to which the transaction database is attached.

VOLSER: The volume serial number of the shared directory to which the transaction database is attached.

DSN: The dataset name of the transaction database.

VOLSER: The volume serial of the transaction database.

AM (ACCESS METHOD): The means by which the transaction database is accessed — either VSAM or BSAM (sequential).

BLKSIZE: The size of a block available to store the transaction report. For VSAM files, this number is 7 bytes less than the CISIZE of the file. This will be the value in the MAXRECLN and AVGRECLN parameters on the cluster definition.

TYPE: Indicates the type of transaction database. In this instance, the file was formatted for CICS.

REPTCOUNT: The maximum number of members (transaction reports) that the transaction database can contain. This value was specified when the file was formatted.

GROUPCOUNT: The number of blocks of data that can be contained in each group (extent). This value was specified when the file was formatted.

AUTODELETE: The AUTODELETE option that was specified when the file was formatted.

INDEX: Indicates whether the file was formatted to allow indexes to occupy more than one block. This field is useful only for Abend-AID report files. NON-EXT or EXT may appear in this field.

EXTENTS: The largest number of groups (extents) that can be used by a single member.

Other Information

This section of the report lists additional characteristics about the file.

BLOCKS IN FILE: The number of blocks allocated for the file.

ALLOCATION GROUPS: The number of groups available in which a report can be stored. This number is derived by subtracting the number of blocks required by the directory and header record from the total number of blocks in the file and then dividing the result by the GROUPCOUNT value.

USED ALLOCATION GROUPS: The number and percentage of allocation groups that contain data.

BLOCKS/TRACK: For sequential files only, the number of blocks that exist in each track.

FILE FORMATTED WITH RELEASE: The release level of CWFXSDUT that was used to format the file.

REQUIRED MINIMUM DDIO RELEASE: The minimum release level of CWFXSDUT that can be used to manage the file.

CURRENT REPORT NUMBER: The number that will be assigned to the next member written to the file. Report numbers are assigned sequentially.

DIRECTORY ENTRIES/BLOCK: The number of directory entries that will fit per block on the file. This number is dependent on the EXTENTS and BLKSIZE values. As the number of EXTENTS increases, the number of directory entries per block decreases (provided the BLKSIZE does not change).

DIRECTORY BLOCKS: A value derived by dividing the REPTCOUNT used to format the file by the number of directory entries per block.

FILE FORMAT: Indicates whether additional bytes are in the directory.

DDIO EXTENDED DIR SPACE: A value used by Compuware for problem determination.

USER EXTENDED DIR SPACE: A value used by Compuware for problem determination.

Column Headings

The header line for the body of the DIRX report includes the following columns:

TRAN/TERM: The name of the transaction. If followed by (M), the member is manually locked.

REPT NUMBER: The number assigned to this member by CWFXSDUT.

SYSTEM: The CICS APPLID.

CODE: The abend code that was returned when the transaction failed.

DATE: The date that the transaction was performed.

TIME OF DAY: The time of day that the transaction was performed.

PROGRAM: The name of the program.

NODE: The node at which the transaction abended.

EXTENTS: The group numbers occupied by this portion of the report.

TOTAL GROUPS: The total number of allocation groups occupied by the transaction report.

Member Summary

The following information appears at the end of the report.

NO OF MEMBERS: The total number of members in the transaction database.

NO OF LOCKED MEMBERS: The total number of locked members in the transaction database.

NO OF UNLOCKED MEMBERS: The total number of unlocked members in the transaction database.

NO OF INCOMPLETE MEMBERS: The total number of members in the transaction database that are missing data and therefore cannot be viewed. Incomplete members can result when a file full condition exists while the member is being written to the file or when certain system failures occur.

LARGEST MEMBER SIZE (IN GROUPS): The size of the largest member in the transaction database, expressed in groups.

SMALLEST MEMBER SIZE (IN GROUPS): The size of the smallest member in the transaction database, expressed in groups.

AVERAGE MEMBER SIZE (IN GROUPS): The size of the average member in the transaction database, expressed in groups.

Space Summary Based on AUTODELETE

The following fields describe the characteristics of the transaction database using the specified AUTODELETE option.

MAXIMUM ALLOCATION GROUPS: The number of allocation groups that can exist in the transaction database.

AVAILABLE ALLOCATION GROUPS: The number of allocation groups occupied by unlocked members plus the number of unused allocation groups.

MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT): The maximum number of members (transaction reports) the transaction database can contain.

AVAILABLE NO OF DIRECTORY ENTRIES: The difference between the maximum number of members the file can contain and the number of locked members.

Shared Directory DIRX Report

When a DIRX is performed on a shared directory, the resulting report includes DIRX information for each transaction database attached to the shared directory for Abend-AID for CICS, or for each Abend-AID report file attached to the shared directory for Abend-AID.

A sample shared directory DIRX report is shown in Figure 2-8 on page 2-37. Ellipses indicate omitted portions.

Figure 2-8. Sample Portion of DIRX Report for a Shared Directory (*Part 1 of 7*)

```
DIRX FOR SHARED DIRECTORY: CX.AA.SDIR
                   VOLSER: PRD929


SHARED DIRECTORY INFORMATION:
AM ----------------------------- VSAM
BLKSIZE ------------------------- 18425
OTHER INFORMATION
  DIRECTORY FORMATTED BY DDIO --- 7.9.0
  REQUIRES MINIMUM DDIO RELEASE - 7.0.0
  TOTAL BLOCKS IN SHARED DIR ---- 135
  USED BLOCKS ------------------- 28(21%)
  NUMBER OF EXTENTS ------------- 1

DIRECTORY CLASS: DATABASE
 CLASS INSTANCE NAME ----------- DATABASE
 CLASS ANCHOR RBN -------------- 12
 NUMBER OF ENTRIES ------------- 4
 CURRENT PUT NUMBER ------------ 108
 CURRENT TRANSACTION NUMBER ---- 1130
 KEY LENGTH -------------------- 50
 KEY OFFSET -------------------- 0
 MAXIMUM ELEMENT LENGTH -------- 150

 DIRECTORY CLASS: LPCNTRL
 CLASS INSTANCE NAME ----------- LPCNTRL
 CLASS ANCHOR RBN -------------- 17
 NUMBER OF ENTRIES ------------- 0
 CURRENT PUT NUMBER ------------ 1
 CURRENT TRANSACTION NUMBER ---- 1
 KEY LENGTH -------------------- 20
 KEY OFFSET -------------------- 0
MAXIMUM ELEMENT LENGTH -------- 32000

 DIRECTORY CLASS: LPDATABS
 CLASS INSTANCE NAME ----------- LPDATABS
 CLASS ANCHOR RBN -------------- 16
 NUMBER OF ENTRIES ------------- 0
 CURRENT PUT NUMBER ------------ 1
 CURRENT TRANSACTION NUMBER ---- 1
 KEY LENGTH -------------------- 44
 KEY OFFSET -------------------- 0
 MAXIMUM ELEMENT LENGTH -------- 150

DIRECTORY CLASS: LPIDRTKN
 CLASS INSTANCE NAME ----------- LPIDRTKN
 CLASS ANCHOR RBN -------------- 15
 NUMBER OF ENTRIES ------------- 0
 CURRENT PUT NUMBER ------------ 1
CURRENT TRANSACTION NUMBER ----- 1
 KEY LENGTH -------------------- 8
 KEY OFFSET -------------------- 0
 MAXIMUM ELEMENT LENGTH -------- 1034
```

Figure 2-9. Sample Portion of DIRX Report for a Shared Directory (*Part 2 of 7*)

```
 DIRECTORY CLASS: LPNUMBER
 CLASS INSTANCE NAME ----------- LPNUMBER
 CLASS ANCHOR RBN -------------- 14
 NUMBER OF ENTRIES ------------- 0
 CURRENT PUT NUMBER ------------ 1
 CURRENT TRANSACTION NUMBER ---- 1
 KEY LENGTH -------------------- 4
 KEY OFFSET -------------------- 0
 MAXIMUM ELEMENT LENGTH -------- 1030

 DIRECTORY CLASS: SHARDDIR
 CLASS INSTANCE NAME ----------- SHARDDIR
CLASS ANCHOR RBN -------------- 9
 NUMBER OF ENTRIES ------------- 103
 CURRENT PUT NUMBER ------------ 937
 CURRENT TRANSACTION NUMBER ---- 9039
 KEY LENGTH -------------------- 96
 KEY OFFSET -------------------- 62
 MAXIMUM ELEMENT LENGTH -------- 32000

 DIRECTORY CLASS: SRCEDIR
 CLASS INSTANCE NAME ----------- SRCEDIR
 CLASS ANCHOR RBN -------------- 13
 NUMBER OF ENTRIES ------------- 0
 CURRENT PUT NUMBER ------------ 1
 CURRENT TRANSACTION NUMBER ---- 1
 KEY LENGTH -------------------- 0
 KEY OFFSET -------------------- 2
 MAXIMUM ELEMENT LENGTH -------- 32000

 DIRECTORY CLASS: CLASS
 CLASS INSTANCE NAME ----------- CLASS
 CLASS ANCHOR RBN -------------- 2
 NUMBER OF ENTRIES ------------- 7
 CURRENT PUT NUMBER ------------ 8
 CURRENT TRANSACTION NUMBER ---- 8
 KEY LENGTH -------------------- 16
 KEY OFFSET -------------------- 0
 MAXIMUM ELEMENT LENGTH -------- 40
```

Figure 2-10. Sample Portion of DIRX Report for a Shared Directory (*Part 3 of 7*) — Abend-AID for CICS only

```
  DIRECTORY OF SHARED DIRECTORY: SJPTXS0.MP2.CAT
                      VOLSER: PRD924
      REPORT  TRAN ABEND        ABEND       ABENDING  ABEND   CICS      CICS      REPORT REPORT DATABASE DATASET NAME
      NUMBER   ID   CODE    DATE      TIME  PROGRAM   OFFSET  JOBNAME   APPLID    STATUS TYPE   (TERMINAL - NET NAME - SMF ID)
------------  ---- ----- ------------------ --------  ------  --------  --------  ------ ----   ------------------------------
         708  TOMH  AEYY OCT 26 2001  14.43 AEYYTMCI  000C58  H01AC184  H01AC184  (L)    TRAN   SJPTXS0.MP2.DBIO1
                         OCT 26 2001  14.43                   H01AC184  H01AC184  DUP           (NONE     -          - CW01  )
                         OCT 26 2001  14.43                   H01AC184  H01AC184  DUP           (NONE     -          - CW01  )
                         OCT 26 2001  14.43                   H01AC184  H01AC184  DUP           (NONE     -          - CW01  )
. . .
. . .
. . .
         705  PYU1  ASCR OCT 21 2001  14.43 ASCRPAYU  000000  H01AC183  H01AC183         TRAN   SJPTXS0.MP2.DBIO2
         699  PYU1  ASCR OCT 21 2001  14.43 ASCRPAYU  000000  H01AC183  H01AC183  (L)    TRAN   SJPTXS0.MP2.DBIO1
         698  PYU1  ASCR OCT 21 2001  14.43 ASCRPAYU  000000  H01AC183  H01AC183         TRAN   SJPTXS0.MP2.DBIO2
. . .
. . .
. . .
                         OCT 21 2001   8.39                   H01AC184  H01AC184  DUP           (%004     - PLAYBACK - CW01  )
                         OCT 21 2001   8.39                   H01AC184  H01AC184  DUP           (%005     - PLAYBACK - CW01  )

                       *** ONLY THE LAST 100 DUPLICATES OF A TOTAL 101 ARE DISPLAYED.


SHARED DIRECTORY ENTRIES SELECTED: 136
```

Figure 2-11. Sample Portion of DIRX Report for a Shared Directory (*Part 3 of 7*) — Abend-AID for CICS only

```
DIRX FOR DATABASE: SJPTXS0.MP2.DBIO1
           VOLSER: PRD934


DATABASE IS CONNECTED
CATALOG DSN: SJPTXS0.MP2.CAT
     VOLSER: PRD924
DSN ----------------------------- SJPTXS0.MP2.DBIO1
VOLSER -------------------------- PRD934
AM(ACCESS METHOD) --------------- BSAM
BLKSIZE ------------------------- 3584
TYPE ---------------------------- CICS
REPTCOUNT ----------------------- 500
GROUPCOUNT ---------------------- 1
AUTODELETE ---------------------- NO
INDEX --------------------------- NON-EXT
EXTENTS ------------------------- 64
OTHER INFORMATION
  BLOCKS IN FILE ---------------- 2925
  ALLOCATION GROUPS ------------- 2891
  USED ALLOCATION GROUPS -------- 1358(47%)
  BLOCKS/TRACK ------------------ 13
  FILE FORMATTED WITH RELEASE --- 7.0.0
  REQUIRED MINIMUM RELEASE ------ 7.0.0
  CURRENT REPORT NUMBER --------- 79
  DIRECTORY ENTRIES/BLOCK ------- 16
  DIRECTORY BLOCKS -------------- 32
  FILE FORMAT ------------------- EXTENDED
  DDIO EXTENDED DIR SPACE ------- 6
  USER EXTENDED DIR SPACE ------- 0

TRAN/TERM  REPT NUMBER  SYSTEM   CODE    DATE       TIME OF DAY  PROGRAM--NODE--EXTENTS
TOMH NONE          708 H01AC184  AEYY  OCT 26 2001  14.43.04.8   TMCICS0H           959   960   963   979   981   989   991   993
                                                                                      995   997   999  1000  1001  1002  1003  1004
                                                                                     1005  1006  1007  1008  1009  1010  1011  1012
                                                                                                                TOTAL GROUPS    24

RLM1 NONE(L)       706 H01AC184  AEYY  OCT 26 2001  14.43.02.4   RLMCICS1           951   952   953   957   962   977   980   982
                                                                                      983   984   985   986   987   988   990   992
                                                                                      994   996   998
                                                                                                                TOTAL GROUPS    19

RLM1 NONE          707 H01AC184  AEYY  OCT 21 2001  14.15.30.6 RLMCICS1             954   955   956   958   961   964   965   966
                                                                                      967   968   969   970   971   972   973   974
                                                                                      975   976   978
                                                                                                                TOTAL GROUPS    19

. . .
. . .
. . .
```

Figure 2-12. Sample Portion of DIRX Report for a Shared Directory (*Part 5 of 7*) — Abend-AID for CICS only

```
SAMP %000          27 H01AC188  ASRA  OCT 20 2001  16.45.21.9 HILL     PLAYBACK     20    21    22    23    24    25    26    27
                                                                                     28    29    30    31    32    33    34    35
                                                                                     36    37
                                                                                                                TOTAL GROUPS    18

SAMP %000          26 H01AC189  ASRA  OCT 20 2001  16.43.21.3 HILL     PLAYBACK      1     2     3     4     5     6     7     8
                                                                                      9    10    11    12    13    14    15    16
                                                                                     17    18    19
                                                                                                                TOTAL GROUPS    19


MEMBER SUMMARY OF SJPTXS0.MP2.DBIO1

NO OF MEMBERS -------------------------------- 68
NO OF LOCKED MEMBERS ------------------------- 5
NO OF UNLOCKED MEMBERS ----------------------- 63
NO OF INCOMPLETE MEMBERS --------------------- 0
LARGEST MEMBER SIZE (IN GROUPS) -------------- 25
SMALLEST MEMBER SIZE (IN GROUPS) ------------- 17
AVERAGE MEMBER SIZE (IN GROUPS) -------------- 19.9
SPACE SUMMARY BASED ON AUTODELETE=NO
MAXIMUM ALLOCATION GROUPS -------------------- 2891
AVAILABLE ALLOCATION GROUPS ------------------ 1533
MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 500
AVAILABLE NO OF DIRECTORY ENTRIES ------------ 432
```

Figure 2-13. Sample Portion of DIRX Report for a Shared Directory (*Part 6 of 7*) — Abend-AID for CICS only

```
DIRX FOR DATABASE: SJPTXS0.MP2.DBIO2
           VOLSER: PRD905


DATABASE IS CONNECTED
CATALOG DSN: SJPTXS0.MP2.CAT
     VOLSER: PRD924
DSN ----------------------------- SJPTXS0.MP2.DBIO2
VOLSER -------------------------- PRD907
AM(ACCESS METHOD) --------------- BSAM
BLKSIZE ------------------------- 3584
TYPE ---------------------------- CICS
REPTCOUNT ----------------------- 500
GROUPCOUNT ---------------------- 1
AUTODELETE ---------------------- NO
INDEX --------------------------- NON-EXT
EXTENTS ------------------------- 64
OTHER INFORMATION
  BLOCKS IN FILE ---------------- 2925
  ALLOCATION GROUPS ------------- 2891
  USED ALLOCATION GROUPS -------- 1397(48%)
  BLOCKS/TRACK ------------------ 13
  FILE FORMATTED WITH RELEASE --- 7.0.0
  REQUIRED MINIMUM RELEASE ------ 7.0.0
  CURRENT REPORT NUMBER --------- 69
  DIRECTORY ENTRIES/BLOCK ------- 16
  DIRECTORY BLOCKS -------------- 32
  FILE FORMAT ------------------- EXTENDED
  DDIO EXTENDED DIR SPACE ------- 6
  USER EXTENDED DIR SPACE ------- 0

TRAN/TERM  REPT NUMBER  SYSTEM   CODE    DATE       TIME OF DAY  PROGRAM--NODE--EXTENTS
RLM1 A243          709 H01AC184  AEI1  OCT 21 2001  14.15.59.5 RLMCICS1 TFHA243    987   988   989   990   991   992   993   994
                                                                                     995   996   997   998   999  1000  1001  1002
                                                                                    1003  1004  1005
                                                                                                                TOTAL GROUPS   19


PYU1 NONE          705 H01AC183  ASCR  OCT 21 2001  14.14.09.9 PAYUPAA1            970   971   972   973   974   975   976   977
                                                                                     978   979   980   981   982   983   984   985
                                                                                     986
                                                                                                                TOTAL GROUPS   17

. . .
. . .
. . .

SAMP %000           34 H01AC189  AEI0  OCT 20 2001  16.45.29.7 SAMPLES  PLAYBACK   151   152   153   154   155   156   157   158
                                                                                     159   160   161   162   163   164   165   166
                                                                                     167   168   169   170   171   172   173
                                                                                                                TOTAL GROUPS   23
```

Figure 2-14. Sample Portion of DIRX Report for a Shared Directory (*Part 7 of 7*) — Abend-AID for CICS only

```
SAMP %001          57 H01AC189  ASRA  OCT 20 2001  16.43.34.8 SAMPLES  PLAYBACK   576   577   578   579   580   581   582   583

                                                                                   584   585   586   587   588   589   590   591
                                                                                   592   593   594   595
                                                                                                       TOTAL GROUPS    20


MEMBER SUMMARY OF SJPTXS0.MP2.DBIO2
NO OF MEMBERS ------------------------------- 68
NO OF LOCKED MEMBERS ------------------------- 12
NO OF UNLOCKED MEMBERS ----------------------- 56
NO OF INCOMPLETE MEMBERS --------------------- 0
LARGEST MEMBER SIZE (IN GROUPS) -------------- 25
SMALLEST MEMBER SIZE (IN GROUPS) ------------- 17
AVERAGE MEMBER SIZE (IN GROUPS) -------------- 20.5
SPACE SUMMARY BASED ON AUTODELETE=NO
MAXIMUM ALLOCATION GROUPS -------------------- 2891
AVAILABLE ALLOCATION GROUPS ------------------ 1494
MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 500
AVAILABLE NO OF DIRECTORY ENTRIES ------------ 432


*  COMMENT COMMENT
*** ALL FUNCTIONS SUCCESSFULLY COMPLETED
```

Figure 2-15. Sample Portion of DIRX Report for an Abend-AID Shared Directory Member List

```
Abend-AID shared directory member list - batch report:

DIRECTORY OF SHARED DIRECTORY: CX.AA.SDIR
                       VOLSER: PRD929

      REPORT  JOBNAME  ABEND       ABEND        ABENDING  ABEND   REPORT REPORT DATABASE DATASET NAME
      NUMBER            CODE    DATE    TIME    PROGRAM   OFFSET  STATUS TYPE
------------  -------- -----  ---------- -----  --------  ------  ------ ----   ---------------------------------------------
         936  AA94ASMD  SOC1 18 APR 2002 17.50  TEST      00009C         JOB    PFHWFB0.AA.DB1
         935  AA94COBD  SOC7 18 APR 2002 15.28  MLECOB    0005BE         JOB    PFHWFB0.AA.DB2
         934  AA94ASMD  SOC1 18 APR 2002 15.28  TEST      00009C  (I)    JOB    PFHWFB0.DUMP.DDIO.DB
         851  AA94ASMD  SOC1 12 APR 2002 14.46  TEST      00009C         JOB    PFHWFB0.AA.DB1
         850  AA94ASMD  SOC1 12 APR 2002 14.45  TEST      00009C         JOB    PFHWFB0.AA.DB2
         849  AA94ASMD  SOC1 12 APR 2002 14.45  TEST      00009C         JOB    PFHWFB0.AA.DB1
         848  AA94COBD  SOC7 12 APR 2002 14.44  MLECOB    0005BE         JOB    PFHWFB0.AA.DB2
         847  AA94ASMD  SOC1 12 APR 2002 14.44  TEST      00009C         JOB    PFHWFB0.AA.DB1
         846  AA94COBD  SOC7 12 APR 2002 14.44  MLECOB    0005BE         JOB    PFHWFB0.AA.DB2
         845  AA94COBD  SOC7 12 APR 2002 14.44  MLECOB    0005BE         JOB    PFHWFB0.AA.DB1
         844  AA94ASMD  SOC1 12 APR 2002 14.44  TEST      00009C         JOB    PFHWFB0.AA.DB2
         843  AA94COBD  SOC7 12 APR 2002 14.44  MLECOB    0005BE         JOB    PFHWFB0.AA.DB1
863  TSONAPR1  SOC7 18 NOV 2041 10.42            000000  (M)    JOB    PFHWFB0.DUMP.DDIO.DB
         857  PFHWFBLC  SOC9 18 NOV 2041 10.42            000000  (L)    JOB    PFHWFB0.DUMP.DDIO.DB
         854  PFHWFBAR  SOC9 18 NOV 2041 10.42            000000  (L)    JOB    PFHWFB0.DUMP.DDIO.DB


SHARED DIRECTORY ENTRIES SELECTED: 99
```

Figure 2-16. Sample Portion of DIRX Report for an Abend-AID Database (*Part 1 of 2*)

```
DATABASE IS CONNECTED
CATALOG DSN: CX.AA.SDIR
     VOLSER: PRD929

DSN ----------------------------- CX.AA.DB1
VOLSER -------------------------- PRD962
AM(ACCESS METHOD) --------------- VSAM
BLKSIZE ------------------------- 18425
TYPE ---------------------------- ABEND-AID
REPTCOUNT ----------------------- 100
GROUPCOUNT ---------------------- 3
AUTODELETE ---------------------- YES
INDEX --------------------------- EXT
EXTENTS ------------------------- 64
OTHER INFORMATION
  BLOCKS IN FILE ---------------- 225
  ALLOCATION GROUPS ------------- 73
  USED ALLOCATION GROUPS -------- 18(25%)
  FILE FORMATTED BY DDIO RELEASE- 7.8.0
  REQUIRED MINIMUM DDIO RELEASE - 7.0.0
  CURRENT REPORT NUMBER --------- 937
  DIRECTORY ENTRIES/BLOCK ------- 83
  DIRECTORY BLOCKS -------------- 2
  FILE FORMAT ------------------- EXTENDED
  DDIO EXTENDED DIR SPACE ------- 6
  USER EXTENDED DIR SPACE ------- 0


JOBNAME    REPT NUMBER JESID   CODE        DATE    TIME OF DAY  DESC     SIZE(K) EXTENTS
AA94ASMD           936 J20618  SOC1     18 APR 2002 17.50.37.6  AA 9.4 SD    162    16    17    18
                                                                                                  TOTAL GROUPS     3

AA94ASMD           851 J24439  SOC1     12 APR 2002 14.46.48.6  AA 9.4 SD    162    13    14    15
                                                                                                  TOTAL GROUPS     3

AA94ASMD           849 J24435  SOC1     12 APR 2002 14.45.21.6  AA 9.4 SD    162    10    11    12
                                                                                                  TOTAL GROUPS     3

AA94ASMD           847 J24433  SOC1     12 APR 2002 14.44.50.2  AA 9.4 SD    162     7     8     9
                                                                                                  TOTAL GROUPS     3
```

Figure 2-17. Sample Portion of DIRX Report for an Abend-AID Database (*Part 2 of 2*)

```
MEMBER SUMMARY OF CX.AA.DB1

NO OF MEMBERS ------------------------------- 8
NO OF LOCKED MEMBERS ------------------------- 0
NO OF UNLOCKED MEMBERS ----------------------- 8
NO OF INCOMPLETE MEMBERS --------------------- 0
LARGEST MEMBER SIZE (IN GROUPS) -------------- 3
SMALLEST MEMBER SIZE (IN GROUPS) ------------- 1
AVERAGE MEMBER SIZE (IN GROUPS) -------------- 2.2

SPACE SUMMARY BASED ON AUTODELETE=YES
MAXIMUM ALLOCATION GROUPS -------------------- 73
AVAILABLE ALLOCATION GROUPS ------------------ 73
MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 100
AVAILABLE NO OF DIRECTORY ENTRIES ------------ 100
```

Shared Directory DIRX Report Fields

The following is a description of the Shared Directory DIRX Report fields.

DIRX FOR SHARED DIRECTORY: The dataset name of the shared directory.

VOLSER: The volume serial number of the shared directory.

SHARED DIRECTORY INFORMATION: Allocation and formatting information about the shared directory.

AM: The means by which the file is accessed — shared directories must be created as VSAM files.

BLKSIZE: The size of a block available to store an entry. For VSAM files, this number is 7 bytes less than the CISIZE of the file. This will be the value in the MAXRECLN and AVGRECLN parameters on the cluster definition.

Other Information

This section of the report lists additional characteristics about the file.

DIRECTORY FORMATTED BY DDIO: The release level of CWFXSDUT or CWAASDUT that was used to format the file.

REQUIRES MINIMUM DDIO RELEASE: The minimum release level of CWFXSDUT or CWAASDUT that can be used to manage the file.

TOTAL BLOCKS IN SHARED DIR: The number of blocks allocated for the shared directory.

USED BLOCKS: The number and percentage of blocks allocated for the shared directory that contain data. These blocks may or may not be full.

NUMBER OF EXTENTS: The number of allocation groups occupied by the report.

The next several subsections of the report contain class information. A separate subsection is included for each class of object. The class information is normally used by Compuware Corporation for problem determination.

DIRECTORY CLASS: The class of object for this instance. (DATABASE, SHARDDIR, and CLASS are the classes used by CWFXSDUT and CWAASDUT.)

CLASS INSTANCE NAME: The name of the class associated with this particular subsection's information (i.e., DATABASE).

CLASS ANCHOR RBN: The relative block number of the root element.

NUMBER OF ENTRIES: The number of records in this instance. For class DATABASE, this should be the number of attached transaction databases. For class SHARDDIR, this should be the total number of entries across all attached databases.

CURRENT PUT NUMBER: The number of the next record to be added to this instance. For class SHARDDIR, this normally will be the next report number assigned.

CURRENT TRANSACTION NUMBER: The next available transaction number.

KEY LENGTH: The length of the key used to access entries in this class.

KEY OFFSET: The position in the record at which the key begins.

MAXIMUM ELEMENT LENGTH: The maximum length of which any element in this instance can have.

The next section of the DIRX report contains a DIRECTORY report for the shared directory. The following is a summary of the DIRECTORY report headings:

DIRECTORY OF SHARED DIRECTORY: The dataset name of the shared directory.

VOLSER: The volume serial number of the shared directory to which the transaction database or Abend-AID report file is attached.

REPORT NUMBER: The number of the transaction report or Abend-AID report.

TRAN ID: The CICS transaction ID (Abend-AID for CICS only).

JOBNAME: The name from the JOB card. If followed by (M), the member is manually locked and will not be automatically deleted when the file becomes full.

ABEND CODE: The CICS transaction abend code for Abend-AID for CICS; the abend code returned when the job failed for Abend-AID.

(ABEND) DATE: The date on which the abend occurred.

(ABEND) TIME: The time at which the abend occurred.

ABENDING PROGRAM: The name of the program in which the abend was detected.

ABEND OFFSET: The position in the record at which the abend occurred.

CICS JOBNAME: The name of the job submitted to bring up CICS (Abend-AID for CICS only).

CICS APPLID: The ID of the CICS region (Abend-AID for CICS only).

REPORT STATUS: Indicates whether the report is locked or unlocked and complete or incomplete. If (L) appears here, the report is locked. If (I) appears here, the report is incomplete. A blank indicates that the report is unlocked and complete. If the word DUP appears in this field, this indicates that a duplicate abend occurred.

REPORT TYPE: Indicates whether the report was a system report or a transaction report for Abend-AID for CICS, or 'JOB' if Abend-AID.

DATABASE DATASET NAME: The name of the dataset containing the entries.

SHARED DIRECTORY ENTRIES SELECTED: This field appears at the end of the DIRECTORY report, and lists the number of transaction reports (entries) included in the report.

The final section of the report contains a DIRX report for each database (transaction database or Abend-AID report file) attached to the shared directory. For a description of the fields included in a DIRX report for a transaction database, see "Transaction Database DIRX Report Fields" on page 2-33. For a description of the fields included in a DIRX report for an Abend-AID report file, see "Abend-AID Report File DIRX Report" on page 2-24.

Source Listing Database DIRX Report

A sample DIRX report for a Source Listing Database is shown in Figure 2-18 on page 2-50.

Figure 2-18. Sample DIRX Report for a Source Listing Database. (*Part 1 of 2*)

```
 DIRX FOR DATABASE: USERID.R770.SO.ADDUPS.DDIO1
           VOLSER: PRD934


DATABASE IS CONNECTED
CATALOG DSN: USERID.R770.SO.ADDUPS.SHRDIR
     VOLSER: PRD917

DSN ---------------------------- USERID.R770.SO.ADDUPS.DDIO1
VOLSER -------------------------- PRD934
AM(ACCESS METHOD) --------------- BSAM
BLKSIZE ------------------------- 18452
TYPE ---------------------------- SOURCE
REPTCOUNT ----------------------- 50
GROUPCOUNT ---------------------- 1
AUTODELETE ---------------------- DUPS
INDEX --------------------------- EXT
EXTENTS ------------------------- 460
OTHER INFORMATION
  BLOCKS IN FILE ---------------- 75
  ALLOCATION GROUPS ------------- 70
  USED ALLOCATION GROUPS -------- 60(86%)
  BLOCKS/TRACK ------------------ 3
  FILE FORMATTED BY DDIO RELEASE- 7.7.0
  REQUIRED MINIMUM DDIO RELEASE - 7.0.0
  CURRENT REPORT NUMBER --------- 13
  DIRECTORY ENTRIES/BLOCK ------- 18
  DIRECTORY BLOCKS -------------- 3
  FILE FORMAT ------------------- EXTENDED
  DDIO EXTENDED DIR SPACE ------- 6
  USER EXTENDED DIR SPACE ------- 0

PROGRAM    LIST NUMBER   COMP DATE    TIME   RC  LANGUAGE SIZE(K)              GROUPS
COBABND2 (L)        12  30 MAY 2000 09:23:43 04 COBIIR4       36                 59    60
                                                                                                  TOTAL GROUPS     2

ALongNa* (L)         7  26 MAY 2000 16.00.55 00 C/ESA V3R2   793                 15    16    17    18    19    20    21    22
ALongNameofatLeast30Characters_10                                                23    24    25    26    27    28    29    30
                                                                                 31    32    33    34    35    36    37    38
                                                                                 39    40    41    42    43    44    45    46
                                                                                 47    48    49    50    51    52    53    54
                                                                                 55    56    57    58
                                                                                                  TOTAL GROUPS    44

COBABEND (L)         6  26 MAY 2000 16:00:28 04 COBIIR4       36                 13    14
                                                                                                  TOTAL GROUPS     2

COBABND1 (L)         5  26 MAY 2000 15:59:13 04 COBIIR4       36                 11    12
                                                                                                  TOTAL GROUPS     2
```

Figure 2-19. Sample DIRX Report for a Source Listing Database. (*Part 2 of 2*)

```
ASMTEST              3  26 MAY 2000 15.57    00 ASMH          72                    3     4     5     6
                                                                                                          TOTAL GROUPS     4

COBABEND             2  26 MAY 2000 15:56:59 04 COBIIR4       36                    1     2
                                                                                                          TOTAL GROUPS     2


MEMBER SUMMARY OF USERID.R770.SO.ADDUPS.DDIO1

NO OF MEMBERS -------------------------------- 7
NO OF LOCKED MEMBERS ------------------------- 5
NO OF UNLOCKED MEMBERS ----------------------- 2
NO OF INCOMPLETE MEMBERS --------------------- 0
LARGEST MEMBER SIZE (IN GROUPS) -------------- 44
SMALLEST MEMBER SIZE (IN GROUPS) ------------- 2
AVERAGE MEMBER SIZE (IN GROUPS) -------------- 8.5

SPACE SUMMARY BASED ON AUTODELETE=DUPS
MAXIMUM ALLOCATION GROUPS -------------------- 70
AVAILABLE ALLOCATION GROUPS ------------------ 16
MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 50
AVAILABLE NO OF DIRECTORY ENTRIES ------------ 45


*** ALL FUNCTIONS SUCCESSFULLY COMPLETED
```

Source Listing Database DIRX Report Fields

The following is a description of the Source Listing Database DIRX report fields.

DIRX FOR DATABASE: The dataset name of the Source Listing Database.

VOLSER: The volume serial number of the Source Listing Database.

DATABASE IS CONNECTED: Indicates the Source Listing Database is attached to a shared directory.

CATALOG DSN: The dataset name of the shared directory to which the Source Listing Database is attached.

VOLSER: The volume serial number of the shared directory to which the Source Listing Database is attached.

DSN: The dataset name of the Source Listing Database

VOLSER: The volume serial number of the Source Listing Database.

AM (ACCESS METHOD): The means by which the Source Listing Database is accessed — either VSAM or BSAM (sequential).

BLKSIZE: The size of a block available to store the source listing. For VSAM files, this number is seven bytes less than the CISIZE of the file. This will be the value in the MAXRECLN and AVGRECLN parameters on the cluster definition.

TYPE: Indicates the type of file format. In this instance the file was formatted for source.

REPTCOUNT: The maximum number of members (source listings) that the Source Listing Database can contain. This value was specified when the file was formatted.

GROUPCOUNT: The number of blocks of data that can be contained in each group (extent). This value was specified when the file was formatted.

AUTODELETE: The AUTODELETE option that was specified when the file was formatted.

INDEX: Indicates whether the file was formatted to allow indexes to occupy more than one block. EXT should appear in this field.

EXTENTS: The largest number of groups (extents) that can be used by a single member.

Other Information

This section of the report lists additional characteristics about the file.

BLOCKS IN FILE: The number of blocks allocated for the file.

ALLOCATION GROUPS: The number of groups available in which a listing can be stored. This number is derived by subtracting the number of blocks required by the directory and header record from the total number of blocks in the file and then dividing the result by the GROUPCOUNT value.

USED ALLOCATION GROUPS: The number, and percentage of allocation groups that contain data.

BLOCKS/TRACK: For sequential files only, the number of blocks that exist in each track.

FILE FORMATTED BY DDIO RELEASE: The release level of CWDDLPUT that was used to format the file.

REQUIRED MINIMUM DDIO RELEASE: The minimum release level of CWDDLPUT that can be used to manage the file.

CURRENT REPORT NUMBER: The highest source listing number on this database, plus 1.

DIRECTORY ENTRIES/BLOCK: The number of directory entries that will fit per block on the file. This number is dependent on the EXTENTS and BLKSIZE values. As EXTENTS increase, the number of directory entries per block decrease (provided the BLKSIZE does not change).

DIRECTORY BLOCKS: A value derived by dividing the REPTCOUNT used to format the file by the number of directory entries per block.

FILE FORMAT: Indicates whether additional bytes are in the directory.

DDIO EXTENDED DIR SPACE: A value used by Compuware for problem determination.

USER EXTENDED DIR SPACE: A value used by Compuware for problem determination.

Column Headings

The header line for the body of the DIRX report includes the following columns:

PROGRAM: The name of the program. If followed by (L), the member is automatically locked. If followed by (M), the member is manually locked. If the name is a LONGNAME (greater than eight characters), an asterisk will be placed in the last position and the full long name will be contained on the next lines.

LIST NUMBER: The number assigned to this member.

COMP DATE: The date the program was compiled.

TIME: The time of day the program was compiled.

RC: The return code from the step that created the member.

LANGUAGE: The program language compiler and version. The following codes can be found in the LANGUAGE column:

ASMF	Assembler F
ASMH	Assembler H
HLASM	High Level Assembler 1.1
HLASM R2	High Level Assembler 1.2
HLASM R3	High Level Assembler 1.3
HLASM R4	High Level Assembler 1.4
OSVSCOB	OS/VS COBOL
OSVSCOBOPT	OS/VS COBOL/CA-Optimizer
C/ESA V3R2	“C” for ESA V3R2
C/390 V1R2	OS/390 “C” V1R2
C/390 V1R3	OS/390 “C” V1R3
C/390 V2R4	OS/390 “C” V2R4
C/390 V2R5	OS/390 “C” V2R5
C/390 V2R6	OS/390 “C” V2R6
C/390 V2R9	OS/390 “C” V2R9
C/390 V2RA	OS/390 “C” V2R10
COBIIR2	VS COBOL II Release 2
COBIIR2OPT	VS COBOL II Release 2/CA-Optimizer II
COBIIR3	VS COBOL II Release 3
COBIIR3OPT	VS COBOL II Release 3/CA-Optimizer II

COBIIR4	VS COBOL II Release 4
COBIIR4 JA	VS COBOL II Release 4 with Japanese Language Support
COBIIR4OPT	VS COBOL II Release 4/CA-Optimizer II
COBII JA	VS COBOL II with Japanese Language Support
COBOL/370	IBM SAA AD/Cycle COBOL/370
COBOL/370 JA	IBM SAA AD/Cycle COBOL/370 with Japanese Language Support
COBOL/370OPT	IBM SAA AD/Cycle COBOL/370/CA-Optimizer II
COBOL/390	COBOL for OS/390 and VM
COBOL/390M	COBOL for OS/390 and VM with MLE (Millennium Language Extensions)
COBOL/MVSM	COBOL for MVS and VM with MLE
COB/370JA	COBOL/370 Japanese Language Support
COB/390OPM	COBOL/390 MLE CA-Optimizer
COB/390OPT	COBOL/390 CA-Optimizer
COB/MVS	COBOL for MVS and VM
COB/MVS JA	COBOL for MVS and VM with Japanese Language Support
COB/MVSOPM	COBOL/MVS MLE CA-Optimizer
COB/MVSOPT	COBOL/MVS CA-Optimizer
COBOL Z/OS	Enterprise COBOL for z/OS and OS/390.
COBOL Z/OSJA	Enterprise COBOL for z/OS and OS/390 with Japanese language support.
COB/Z V3R2	Enterprise COBOL for z/OS and OS/390 Release 3.2
COB Z/OSJA	Enterprise COBOL for z/OS and OS/390 Release 3.2 with Japanese Language Support
PL/I V1R5	PL/I V1
PL/I V2	PL/I V2
PL/I AD V1	PL/I for MVS and VM
PL1VA V2R2	PL/I VisualAge V2R2
PL1EN V3R1	Enterprise PL/I for z/OS
PL1EN V3R2	Enterprise PL/I for z/OS

SIZE(K): The number of KBs this member occupies on the Source Listing Database.

GROUPS: The allocation group numbers occupied by the source listing.

TOTAL GROUPS: The total number of allocation groups occupied by the source listing.

Member Summary

The following information appears at the end of the report.

NO OF MEMBERS: The total number of members in the Source Listing Database.

NO OF LOCKED MEMBERS: The total number of locked members in the Source Listing Database.

NO OF UNLOCKED MEMBERS: The total number of unlocked members in the Source Listing Database.

NO OF INCOMPLETE MEMBERS: The total number of members in the Source Listing Database that are missing data and therefore cannot be viewed. Incomplete members can

result when a file full condition exists while the member is being written to the file (and dynamic database creation is not in effect), or when certain system failures occur.

LARGEST MEMBER SIZE (IN GROUPS): The size of the largest member in the Source Listing Database, expressed in groups.

SMALLEST MEMBER SIZE (IN GROUPS): The size of the smallest member in the Source Listing Database, expressed in groups.

AVERAGE MEMBER SIZE (IN GROUPS): The size of the average member in the Source Listing Database, expressed in groups.

Space Summary Based on AUTODELETE

The following fields describe the characteristics of the Source Listing Database using the specified AUTODELETE option.

MAXIMUM ALLOCATION GROUPS: The number of allocation groups that can exist in the Source Listing Database.

AVAILABLE ALLOCATION GROUPS: The number of allocation groups occupied by unlocked members plus the number of unused allocation groups.

MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT): The maximum number of members (source listings) the Source Listing Database can contain.

AVAILABLE NO OF DIRECTORY ENTRIES: The difference between the maximum number of members the file can contain and the number of locked (manually locked when AUTODELETE=STAGED) members.

Source Listing Shared Directory (SRCEDIR) DIRX Report

When a DIRX is performed on a shared directory, the resulting report includes DIRX information for each Source Listing Database attached to the shared directory.

A sample DIRX report for a Source Listing Shared Directory is shown in Figure 2-20 on page 2-57.

Figure 2-20. Sample DIRX Report for a Source Listing Shared Directory. (*Part 1 of 6*)

```
DIRX FOR SHARED DIRECTORY: USERID.R770.SO.ADDUPS.SHRDIR
                   VOLSER: PRD917


SHARED DIRECTORY INFORMATION:
AM ----------------------------- VSAM
BLKSIZE ------------------------- 4089
OTHER INFORMATION
  DIRECTORY FORMATTED BY DDIO --- 7.7.0
  REQUIRES MINIMUM DDIO RELEASE - 7.0.0
  TOTAL BLOCKS IN SHARED DIR ---- 180
  USED BLOCKS -------------------- 27(15%)
  NUMBER OF EXTENTS ------------- 1


DIRECTORY CLASS: DATABASE
 CLASS INSTANCE NAME ----------- DATABASE
 CLASS ANCHOR RBN -------------- 12
 NUMBER OF ENTRIES ------------- 0
 CURRENT PUT NUMBER ------------ 1
 CURRENT TRANSACTION NUMBER ---- 1
 KEY LENGTH -------------------- 50
 KEY OFFSET -------------------- 0
 MAXIMUM ELEMENT LENGTH -------- 150


 DIRECTORY CLASS: LPCNTRL
 CLASS INSTANCE NAME ----------- LPCNTRL
 CLASS ANCHOR RBN -------------- 17
 NUMBER OF ENTRIES ------------- 1
 CURRENT PUT NUMBER ------------ 2
 CURRENT TRANSACTION NUMBER ---- 13
 KEY LENGTH -------------------- 20
 KEY OFFSET -------------------- 0
 MAXIMUM ELEMENT LENGTH -------- 32000


DIRECTORY CLASS: LPDATABS
 CLASS INSTANCE NAME ----------- LPDATABS
 CLASS ANCHOR RBN -------------- 16
 NUMBER OF ENTRIES ------------- 2
 CURRENT PUT NUMBER ------------ 4
 CURRENT TRANSACTION NUMBER ---- 42
 KEY LENGTH -------------------- 44
 KEY OFFSET -------------------- 0
 MAXIMUM ELEMENT LENGTH -------- 150
```

Figure 2-21. Sample DIRX Report for a Source Listing Shared Directory. (*Part 2 of 6*)

```
DIRECTORY CLASS: LPIDRTKN
CLASS INSTANCE NAME ----------- LPIDRTKN
CLASS ANCHOR RBN -------------- 15
NUMBER OF ENTRIES ------------- 12
CURRENT PUT NUMBER ------------ 13
CURRENT TRANSACTION NUMBER ---- 13
KEY LENGTH -------------------- 8
KEY OFFSET -------------------- 0
MAXIMUM ELEMENT LENGTH -------- 1034


DIRECTORY CLASS: LPNUMBER
CLASS INSTANCE NAME ----------- LPNUMBER
CLASS ANCHOR RBN -------------- 14
NUMBER OF ENTRIES ------------- 12
CURRENT PUT NUMBER ------------ 13
CURRENT TRANSACTION NUMBER ---- 13
KEY LENGTH -------------------- 4
KEY OFFSET -------------------- 0
MAXIMUM ELEMENT LENGTH -------- 1030


DIRECTORY CLASS: SHARDDIR
CLASS INSTANCE NAME ----------- SHARDDIR
CLASS ANCHOR RBN -------------- 9
NUMBER OF ENTRIES ------------- 0
CURRENT PUT NUMBER ------------ 1
CURRENT TRANSACTION NUMBER ---- 1
KEY LENGTH -------------------- 96
KEY OFFSET -------------------- 62
MAXIMUM ELEMENT LENGTH -------- 32000


DIRECTORY CLASS: SRCEDIR
CLASS INSTANCE NAME ----------- SRCEDIR
CLASS ANCHOR RBN -------------- 13
NUMBER OF ENTRIES ------------- 7
CURRENT PUT NUMBER ------------ 8
CURRENT TRANSACTION NUMBER ---- 73
KEY LENGTH -------------------- 0
KEY OFFSET -------------------- 2
MAXIMUM ELEMENT LENGTH -------- 32000


DIRECTORY CLASS: CLASS
CLASS INSTANCE NAME ----------- CLASS
CLASS ANCHOR RBN -------------- 2
NUMBER OF ENTRIES ------------- 7
CURRENT PUT NUMBER ------------ 8
CURRENT TRANSACTION NUMBER ---- 8
KEY LENGTH -------------------- 16
KEY OFFSET -------------------- 0
MAXIMUM ELEMENT LENGTH -------- 40
```

Figure 2-22. Sample DIRX Report for a Source Listing Shared Directory. (*Part 3 of 6*)

```
 DIRECTORY OF SHARED DIRECTORY: USERID.R770.SO.ADDUPS.SHRDIR
                       VOLSER: PRD917

                 REPORT
 PROGRAM         NUMBER  STATUS    COMPILE DATE   TIME   RC  LANGUAGE    SIZE(K)  DATABASE DATASET NAME
--------  ------------  --------- ------------ -------- --  ----------  -------  ---------------------------------------------
COBABND2            12  (L)       30 MAY 2000  09:23:43 04  COBIIR4         36   USERID.R770.SO.ADDUPS.DDIO1
COBABND1            11            30 MAY 2000  09:21:30 04  COBIIR4         40   USERID.R770.SO.ADDUPS.DDIO3
COBDEMO2            10            30 MAY 2000  09:18:50 04  COBIIR4        120   USERID.R770.SO.ADDUPS.DDIO3
COBDEMO1             9            30 MAY 2000  09:07:14 04  COBIIR4        120   USERID.R770.SO.ADDUPS.DDIO3
COBDEMO1             8            26 MAY 2000  16:03:17 04  COBIIR4        120   USERID.R770.SO.ADDUPS.DDIO3
ALongNa*             7  (L)       26 MAY 2000  16.00.55 00  C/ESA V3R2     793   USERID.R770.SO.ADDUPS.DDIO1
 ALongNameofatLeast30Characters_10
COBABEND             6  (L)       26 MAY 2000  16:00:28 04  COBIIR4         36   USERID.R770.SO.ADDUPS.DDIO1
COBABND1             5  (L)       26 MAY 2000  15:59:13 04  COBIIR4         36   USERID.R770.SO.ADDUPS.DDIO1
ASMTEST              4  (L)       26 MAY 2000  15.58    00  ASMH            72   USERID.R770.SO.ADDUPS.DDIO1
ASMTEST              3            26 MAY 2000  15.57    00  ASMH            72   USERID.R770.SO.ADDUPS.DDIO1
COBABEND             2            26 MAY 2000  15:56:59 04  COBIIR4         36   USERID.R770.SO.ADDUPS.DDIO1
ALongNa*             1            26 MAY 2000  15.54.23 00  C/ESA V3R2     800   USERID.R770.SO.ADDUPS.DDIO3
 ALongNameofatLeast30Characters_10


SHARED DIRECTORY ENTRIES SELECTED: 12

DIRX FOR DATABASE: USERID.R770.SO.ADDUPS.DDIO1
           VOLSER: PRD934


DATABASE IS CONNECTED
CATALOG DSN: USERID.R770.SO.ADDUPS.SHRDIR
     VOLSER: PRD917

DSN ----------------------------- USERID.R770.SO.ADDUPS.DDIO1
VOLSER -------------------------- PRD934
AM(ACCESS METHOD) --------------- BSAM
BLKSIZE ------------------------- 18452
TYPE ---------------------------- SOURCE
REPTCOUNT ----------------------- 50
GROUPCOUNT ---------------------- 1
AUTODELETE ---------------------- DUPS
INDEX --------------------------- EXT
EXTENTS ------------------------- 460
```

Figure 2-23. Sample DIRX Report for a Source Listing Shared Directory. (*Part 4 of 6*)

```
OTHER INFORMATION
  BLOCKS IN FILE ---------------- 75
  ALLOCATION GROUPS ------------- 70
  USED ALLOCATION GROUPS -------- 60(86%)
  BLOCKS/TRACK ------------------ 3
  FILE FORMATTED BY DDIO RELEASE- 7.7.0
  REQUIRED MINIMUM DDIO RELEASE - 7.0.0
  CURRENT REPORT NUMBER --------- 13
  DIRECTORY ENTRIES/BLOCK ------- 18
  DIRECTORY BLOCKS -------------- 3
  FILE FORMAT ------------------- EXTENDED
  DDIO EXTENDED DIR SPACE ------- 6
  USER EXTENDED DIR SPACE ------- 0

PROGRAM    LIST NUMBER   COMP DATE    TIME   RC  LANGUAGE SIZE(K)              GROUPS
COBABND2 (L)        12  30 MAY 2000 09:23:43 04 COBIIR4       36                 59    60
                                                                                                TOTAL GROUPS     2


ALongNa* (L)         7  26 MAY 2000 16.00.55 00 C/ESA V3R2   793                 15    16    17    18    19    20    21    22
ALongNameofatLeast30Characters_10                                                23    24    25    26    27    28    29    30
                                                                                 31    32    33    34    35    36    37    38
                                                                                 39    40    41    42    43    44    45    46
                                                                                 47    48    49    50    51    52    53    54
                                                                                 55    56    57    58
                                                                                                TOTAL GROUPS    44

COBABEND (L)         6  26 MAY 2000 16:00:28 04 COBIIR4       36                 13    14
                                                                                                TOTAL GROUPS     2

COBABND1 (L)         5  26 MAY 2000 15:59:13 04 COBIIR4       36                 11    12
                                                                                                TOTAL GROUPS     2

ASMTEST  (L)         4  26 MAY 2000 15.58    00 ASMH          72                  7     8     9    10
                                                                                                TOTAL GROUPS     4

ASMTEST              3  26 MAY 2000 15.57    00 ASMH          72                  3     4     5     6
                                                                                                TOTAL GROUPS     4

COBABEND             2  26 MAY 2000 15:56:59 04 COBIIR4       36                  1     2
                                                                                                TOTAL GROUPS     2
```

Figure 2-24. Sample DIRX Report for a Source Listing Shared Directory. (*Part 5 of 6*)

```
MEMBER SUMMARY OF USERID.R770.SO.ADDUPS.DDIO1

NO OF MEMBERS ------------------------------- 7
NO OF LOCKED MEMBERS ------------------------- 5
NO OF UNLOCKED MEMBERS ----------------------- 2
NO OF INCOMPLETE MEMBERS --------------------- 0
LARGEST MEMBER SIZE (IN GROUPS) -------------- 44
SMALLEST MEMBER SIZE (IN GROUPS) ------------- 2
AVERAGE MEMBER SIZE (IN GROUPS) -------------- 8.5

SPACE SUMMARY BASED ON AUTODELETE=DUPS
MAXIMUM ALLOCATION GROUPS -------------------- 70
AVAILABLE ALLOCATION GROUPS ------------------ 16
MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 50
AVAILABLE NO OF DIRECTORY ENTRIES ------------ 45
DIRX FOR DATABASE: USERID.R770.SO.ADDUPS.DDIO3
           VOLSER: PRD969


DATABASE IS CONNECTED
CATALOG DSN: USERID.R770.SO.ADDUPS.SHRDIR
     VOLSER: PRD917

DSN ----------------------------- USERID.R770.SO.ADDUPS.DDIO3
VOLSER -------------------------- PRD969
AM(ACCESS METHOD) --------------- VSAM
BLKSIZE ------------------------- 20473
TYPE ---------------------------- SOURCE
REPTCOUNT ----------------------- 50
GROUPCOUNT ---------------------- 1
AUTODELETE ---------------------- YES
INDEX --------------------------- EXT
EXTENTS ------------------------- 460
OTHER INFORMATION
  BLOCKS IN FILE ---------------- 74
  ALLOCATION GROUPS ------------- 69
USED ALLOCATION GROUPS -------- 60(87%)
  FILE FORMATTED BY DDIO RELEASE- 7.7.0
  REQUIRED MINIMUM DDIO RELEASE - 7.0.0
  CURRENT REPORT NUMBER --------- 12
  DIRECTORY ENTRIES/BLOCK ------- 20
  DIRECTORY BLOCKS -------------- 3
  FILE FORMAT ------------------- EXTENDED
  DDIO EXTENDED DIR SPACE ------- 6
  USER EXTENDED DIR SPACE ------- 0
```

Figure 2-25. Sample DIRX Report for a Source Listing Shared Directory. (*Part 6 of 6*)

```
PROGRAM    LIST NUMBER   COMP DATE    TIME   RC  LANGUAGE SIZE(K)               GROUPS
COBABND1            11  30 MAY 2000 09:21:30 04 COBIIR4       40                   59    60
                                                                                                  TOTAL GROUPS     2

COBDEMO2            10  30 MAY 2000 09:18:50 04 COBIIR4      120                   53    54    55    56    57    58
                                                                                                  TOTAL GROUPS     6

COBDEMO1             9  30 MAY 2000 09:07:14 04 COBIIR4      120                   47    48    49    50    51    52
                                                                                                  TOTAL GROUPS     6

COBDEMO1             8  26 MAY 2000 16:03:17 04 COBIIR4      120                   41    42    43    44    45    46
                                                                                                  TOTAL GROUPS     6

ALongNa*             1  26 MAY 2000 15.54.23 00 C/ESA V3R2   800                    1     2     3     4     5     6     7     8
ALongNameofatLeast30Characters_10                                                    9    10    11    12    13    14    15    16
                                                                                    17    18    19    20    21    22    23    24
                                                                                    25    26    27    28    29    30    31    32
                                                                                    33    34    35    36    37    38    39    40
                                                                                                    TOTAL GROUPS    40


MEMBER SUMMARY OF USERID.R770.SO.ADDUPS.DDIO3

NO OF MEMBERS ------------------------------- 5
NO OF LOCKED MEMBERS ------------------------ 0
NO OF UNLOCKED MEMBERS ----------------------- 5
NO OF INCOMPLETE MEMBERS --------------------- 0
LARGEST MEMBER SIZE (IN GROUPS) -------------- 40
SMALLEST MEMBER SIZE (IN GROUPS) ------------- 2
AVERAGE MEMBER SIZE (IN GROUPS) -------------- 12.0
SPACE SUMMARY BASED ON AUTODELETE=YES
MAXIMUM ALLOCATION GROUPS -------------------- 69
AVAILABLE ALLOCATION GROUPS ------------------ 69
MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 50
AVAILABLE NO OF DIRECTORY ENTRIES ------------ 50


*** ALL FUNCTIONS SUCCESSFULLY COMPLETED


****************************** BOTTOM OF DATA ******************************************************************************
```

Shared Directory DIRX Report Fields

The following is a description of the shared directory DIRX report fields.

DIRX FOR SHARED DIRECTORY: The dataset name of the shared directory.

VOLSER: The volume serial number of the shared directory.

SHARED DIRECTORY INFORMATION: Allocation and formatting information about the shared directory.

AM: The means by which the file is accessed — shared directories must be created as VSAM files.

BLKSIZE: The size of a block available to store an entry. For VSAM files, this number is seven bytes less than the CISIZE of the file. This will be the value in the MAXRECLN and AVGRECLN parameters on the cluster definition.

Other Information

This section of the report lists additional characteristics about the file.

DIRECTORY FORMATTED BY DDIO: The release level of CWDDLPUT that was used to format the file.

REQUIRES MINIMUM DDIO RELEASE: The minimum release level of CWDDLPUT that can be used to manage the file.

TOTAL BLOCKS IN SHARED DIR: The number of blocks allocated for the shared directory.

USED BLOCKS: The number and percentage of blocks allocated for the shared directory that contain data. These blocks may or may not be full.

NUMBER OF EXTENTS: The number of VSAM extents allocated to the file.

The next several subsections of the report contain class information. A separate subsection is included for each class of object. The class information is normally used by Compuware Corporation for problem determination.

DIRECTORY CLASS: The class of object for the instance. (LPDATABS, SRCEDIR, CLASS, LPCNTRL, LPIDRTKN, LPNUMBER are the classes used by LP).

CLASS INSTANCE NAME: The name of the class associated with this particular subsection's information (i.e., LPDATABS).

CLASS ANCHOR RBN: The relative block number of the root element for this class.

NUMBER OF ENTRIES: The number of records in this instance. For class LPDATABS, this should be the number of attached Source Listing Databases. For classes LPNUMBER and LPIDRTRN, this should correspond to the total number of entries across all attached databases.

CURRENT PUT NUMBER: The internal number of the next record to be added to this instance. It does **not** reflect the next report number to be assigned.

CURRENT TRANSACTION NUMBER: The next available transaction number.

KEY LENGTH: The length of the key used to access entries in this class.

KEY OFFSET: The position in the record at which the key begins.

MAXIMUM ELEMENT LENGTH: The maximum length of which any element in this instance can have.

The next section of the DIRX report contains a DIRECTORY report for the shared directory. The following is a summary of the DIRECTORY report headings:

DIRECTORY OF SHARED DIRECTORY: The dataset name of the shared directory.

VOLSER: The volume serial number of the shared directory.

PROGRAM: The name of the program. If the name is a LONGNAME (greater than eight characters), an asterisk will be placed in the last position and the full long name will be contained on the next lines.

REPORT NUMBER: The number assigned to this member.

STATUS: Indicates whether the listing is locked or unlocked, and complete or incomplete. If (L) appears here, the listing is automatically locked. If (M) appears here, the listing is manually locked. If (I) appears here, the listing is incomplete. A blank indicates that the listing is unlocked and complete.

COMPILE DATE: The date the program was compiled.

TIME: The time of day the program was compiled.

RC: the return code from the step that created the member.

LANGUAGE: The program language compiler and version. The following codes can be found in the LANGUAGE column:

ASMF	Assembler F
ASMH	Assembler H
HLASM	High Level Assembler 1.1
HLASM R2	High Level Assembler 1.2
HLASM R3	High Level Assembler 1.3
HLASM R4	High Level Assembler 1.4
OSVSCOB	OS/VS COBOL
OSVSCOBOPT	OS/VS COBOL/CA-Optimizer
C/ESA V3R2	“C” for ESA V3R2
C/390 V1R2	OS/390 “C” V1R2
C/390 V1R3	OS/390 “C” V1R3
C/390 V2R4	OS/390 “C” V2R4
C/390 V2R5	OS/390 “C” V2R5
C/390 V2R6	OS/390 “C” V2R6
C/390 V2R9	OS/390 “C” V2R9
C/390 V2RA	OS/390 “C” V2R10
COBIIR2	VS COBOL II Release 2
COBIIR2OPT	VS COBOL II Release 2/CA-Optimizer II
COBIIR3	VS COBOL II Release 3
COBIIR3OPT	VS COBOL II Release 3/CA-Optimizer II
COBIIR4	VS COBOL II Release 4
COBIIR4 JA	VS COBOL II Release 4 with Japanese Language Support
COBIIR4OPT	VS COBOL II Release 4/CA-Optimizer II
COBII JA	VS COBOL II with Japanese Language Support
COBOL/370	IBM SAA AD/Cycle COBOL/370
COBOL/370 JA	IBM SAA AD/Cycle COBOL/370 with Japanese Language Support
COBOL/370OPT	IBM SAA AD/Cycle COBOL/370/CA-Optimizer II

COBOL/390	COBOL for OS/390 and VM
COBOL/390M	COBOL for OS/390 and VM with MLE (Millennium Language Extensions)
COBOL/MVSM	COBOL for MVS and VM with MLE
COB/370JA	COBOL/370 Japanese Language Support
COB/390OPM	COBOL/390 MLE CA-Optimizer
COB/390OPT	COBOL/390 CA-Optimizer
COB/MVS	COBOL for MVS and VM
COB/MVS JA	COBOL for MVS and VM with Japanese Language Support
COB/MVSOPM	COBOL/MVS MLE CA-Optimizer
COB/MVSOPT	COBOL/MVS CA-Optimizer
COBOL Z/OS	Enterprise COBOL for z/OS and OS/390.
COBOL Z/OSJA	Enterprise COBOL for z/OS and OS/390 with Japanese language support.
COB/Z V3R2	Enterprise COBOL for z/OS and OS/390 Release 3.2
COB Z/OSJA	Enterprise COBOL for z/OS and OS/390 Release 3.2 with Japanese Language Support
PL/I V1R5	PL/I V1
PL/I V2	PL/I V2
PL/I AD V1	PL/I for MVS and VM
PL1VA V2R2	PL/I VisualAge V2R2
PL1EN V3R1	Enterprise PL/I for z/OS
PL1EN V3R2	Enterprise PL/I for z/OS

SIZE(K): The number of KBs this member occupies on the Source Listing Database.

DATABASE DATASET NAME: The name of the dataset containing the entry.

SHARED DIRECTORY ENTRIES SELECTED: This field appears at the end of the DIRECTORY report, and lists the number of source listings (entries) included in the report.

The final section of the report contains a DIRX report for each Source Listing Database attached to the shared directory. For a description of the fields included in a DIRX report for a Source Listing Database, see “Source Listing Database DIRX Report Fields” on page 2-52

Chapter 3.
CSS Utilities

CSS Utilities provides a convenient, menu-driven way to perform CSS Language Processor functions and manipulate DDIO files.

Language Processor Functions

The CSS Language Processor is a set of programs that analyzes the output from a compiler or assembler execution and stores that information in a source listing DDIO file. CSS Utilities Language Processor functions include generating preprocessing and postprocessing JCL.

Note: Preprocessing and postprocessing are described in "CREATEJCL - Preparing Compiler Listings for the LP" on page 3-4.

DDIO File Functions

CSS Utilities enables you to work with two types of DDIO files: source listing files and Abend-AID report files. Both file types are described below.

Source Listing File

A source listing file is a DDIO file that stores source listing members. To create a source listing member, CSS recompiles your program to obtain a compiler listing (or uses an existing compiler listing) and extracts information from particular portions of the listing. The resulting information is then stored as a member in a source listing file.

Source listing files can be the original standard DDIO file format and/or the new shared directory with attached database file formats.

For a description of how source listing files are created and used, see "SOURCE - Working with Source Listing Files" on page 3-10.

Abend-AID Report File

An Abend-AID report file is a DDIO file that stores diagnostic reports. Diagnostic reports are created by Abend-AID when an abend occurs. Each report is stored as a member in the Abend-AID report file.

Abend-AID report files can be the original standard DDIO file format and/or the new shared directory with attached database file formats.

For a description of how Abend-AID report files are created and used, see "AAREPORT - Working with Abend-AID Report Files" on page 3-37.

CSS Utilities DDIO file functions include:

- File creation and formatting
- File deletion
- Member manipulation (for example, locking file members, copying members, importing members, or exporting members to another DDIO file).

- File manipulation (for example, file deletion, verify shared directory, attached database list (for shared directory), change auto-delete characteristics on a database, etc.).

CSS Utilities functions may also be performed by running the following batch utilities:

- CWDDSUTL batch utility for standard DDIO file functions. For detailed information, see **Chapter 4, "Batch File Utility CWDDSUTL"**.
- CWAASDUT batch utility for the report shared directory and database functions. For detailed information, see **Chapter 6, "Batch File Utility CWAASDUT"**.
- CWDDLPUT batch utility for source listing shared directory and database functions. For detailed information, see **Chapter 7, "Batch File Utility CWDDLPUT"**.

DDIO Calculator function:

- The Compuware DDIO Calculator is a tool for calculating your allocation and formatting of DDIO files (source and report) and databases (source, report and transaction). Although an existing file is not required to use this tool, it is highly recommended for use as a model for fine tuning. Otherwise, calculating file size and determining optimal formatting parameters will result in trial and error.

  Note: It is not used for shared directory allocation or tuning. For a description, see "Compuware DDIO Calculator — DDIOCALC" on page 3-61.

Getting Started

You may invoke CSS Utilities differently depending on how it was installed at your site. To verify the installation procedure, see the *ECC Installation and Customization Guide*. If you need additional assistance to determine how to invoke CSS Utilities, see your systems programmer. CSS Utilities may also be invoked from the Compuware Viewing Facility by entering the 'CSSUTIL" command (see "Entry Panel Fields" on page 8-3).

When you invoke CSS Utilities, the CSS Utilities - Primary Menu (Figure 3-1) is displayed.

Figure 3-1. CSS Utilities - Primary Menu

```
----------- Compuware CSS Utilities ---- Primary Menu ----------   CSS V8.0.0
 OPTION  ===>


   1  Settings    - Specify General Settings
   2  CreateJCL   - Create JCL to invoke Compuware Language Processor(LP)
   3  Source      - Create/List/Maintain Source Listing Files/SHRDIR/Databases
   4  AAReport    - Create/List/Maintain Abend-AID Report FilesSHRDIR/DB(s)
   5  DDIOCalc    - Calculate DDIO & Database File Sizes

   L  LP          - Learn about Compuware Language Processor
   D  DDIO        - Learn about Source Listing and Report DDIO Files
   S  SD/Database - Learn about Shared Directory and Database Files
   C  Legal       - Display Full Copyright/Trade Secret Notice

     Copyright  (c)  2002, Compuware Corporation. All rights Reserved.
     Unpublished - rights reserved under the Copyright laws of the United
     States.
                          (800) 538-7822
                       Compuware Shared Services

                     Enter END command to terminate
```

Note: Options 1, 2, 3, and 4 have been renumbered, starting with this release, so that the Create JCL, Source and Abend-AID report numbers will match those on the Viewing Facility main menu.

To select an option, enter the corresponding number at the OPTION prompt. The primary menu options are:

SETTINGS

Displays general (global) CSS Utilities settings. General settings may be overridden by settings on other screens.

CREATEJCL

Enables you to create and store a compiler listing or read and store an existing compiler listing. By selecting this option, you can select whether to create JCL to run the Preprocessor or Postprocessor.

SOURCE

Enables you to set options for creating, reformatting, displaying, and manipulating source listing files and/or source shared directories.

AAREPORT

Enables you to set options for creating, reformatting, displaying, and manipulating Abend-AID report files and/or report shared directories.

DDIOCALC

Invokes the CSS DDIO Calculator tool, described later in this chapter.

LP

Displays an introduction to the Compuware Language Processor.

DDIO

Displays an overview of source listing and Abend-AID report files.

SD/Database

Displays an overview of Shared Directory and Report Database files.

LEGAL

Displays the full CSS Utilities copyright and trade secrets notice.

To get help for a particular CSS Utilities screen, press **PF1**.

SETTINGS - General Settings

CSS Utilities General Settings enable you to configure options used throughout CSS Utilities. To edit the general settings, complete the following steps:

1. Select SETTINGS from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2). The CSS Utilities - General Settings screen (Figure 3-2) is displayed.

   Note: The CSS Utilities - General Settings screen can also be accessed by using the “SEtup” or “SE” command from many of the CSS Utilities screens.

Figure 3-2. CSS Utilities - General Settings

```
----------------- COMPUWARE CSS UTILITIES ---- General Settings ---    V8.0.0
COMMAND ===>

NOTE:  The following settings are used ONLY in the generation of the batch
       utility jobs that are generated from the online utility panels.

Language for panel/message use ===> ENGLISH  (ENGLISH/JAPANESE)

Optional CSS load library to use for STEPLIB in batch job generation:
  Dataset name ===>

Optional XOPTIONS file for XPEDITER/TSO Enhanced FIND:
  Dataset name ===>

Batch Processing Items:
 Hold Queue Sysout Class ===>

Job Statement
===> //JOBNAME  JOB  ('OCXBAS7.6DOC',22),'JOHN SMITH'
===> //*
===> //*
===> //*

            Press ENTER to process or enter END command to terminate
```

2. Edit the settings as needed, using the following table as a guide.

Table 3-1. CSS Utilities General Settings

To:	Do This:
Specify the language to be used for displaying CSS Utilities panels	Enter ENGLISH or JAPANESE in the Language field.
Specify the CSS Load Library DSN to be used as the STEPLIB by JOBS submitted from CSS Utilities or when JCL is created by the create JCL function	Enter a dataset name in the CSS Load Library field.
Specify an XOPTIONS dataset to be used with the XPEDITER/TSO Enhanced FIND function	Enter the XOPTIONS dataset name in the Optional XOPTIONS file for XPEDITER/TSO Enhanced FIND field.
Define the SYSOUT hold output class for batch processing	Enter a class letter in the Hold Queue Sysout Class field. **Note:** You must specify a SYSOUT class if you plan to use the **Preprocessor** command (available on the CREATEJCL menu).
Enter JCL job statement information	Enter a job statement in the Job Statement fields.

3. After you have finished editing the fields, perform the appropriate action:
   - To apply the settings and return to the CSS Utilities - Primary Menu, ensure that the Command prompt is blank and press **Enter**.
   - To cancel the changes and return to the previous screen, enter **END** at the Command prompt.

CREATEJCL - Preparing Compiler Listings for the LP

Source listing files contain one or more source listing reports. Compuware products use source listing reports in a variety of ways:

- to make source data available in Abend-AID reports
- to store application program listings
- to provide source support for interactive debugging with the XPEDITER products
- to merge source data with transaction abend data for online viewing.

CSS uses compiler listings to generate source listing reports. There are two ways to generate a source listing report: preprocessing, and postprocessing.

Preprocessing

Preprocessing performs the following steps:

1. Sets compiler options as necessary.
2. Invokes the compiler and compiles your program to produce a compiler listing.
3. Invokes the postprocessor, discussed in the following section.

Compuware recommends running the preprocessor rather than exclusively running the postprocessor. The preprocessor sets compiler options before invoking the compiler. This ensures that the compiler listing (and the resulting source listing report) contains all required information. Exclusively running the postprocessor forces CSS to use an existing compiler listing that may not contain all required information.

You can run the preprocessor in either of two ways:

- using sample JCL provided with Compuware Shared Services
- using CSS Utilities.

Running the Preprocessor Using Sample JCL

To run the preprocessor using sample JCL provided with CSS, see the Language Processor chapter appropriate for your language:

- **Chapter 9, "COBOL Language Processor"**,
- **Chapter 10, "PL/I Language Processor"**,
- **Chapter 11, "Assembler Language Processor"**, or
- **Chapter 12, "C Language Processor"**.

Running the Preprocessor Using CSS Utilities

To use CSS Utilities to run the preprocessor, complete the following steps:

1. Select CREATEJCL from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2). The CSS Utilities - CREATEJCL Menu (Figure 3-3) is displayed.

Figure 3-3. CSS Utilities - CREATEJCL Menu

```
----------- COMPUWARE CSS UTILITIES ---- Convert JCL Menu    -----      V8.0.0
OPTION ===>


       1  Preprocessor         - Create LP preprocessor JCL
       2  Postprocessor        - Create LP postprocessor JCL


      LP  Learn about Compuware language Processor
      SE  General Settings
      DD  Learn about Compuware DDIO files
      SD  Learn about Compuware Shared Directory and Database files









                 Enter END command to terminate
```

2. Select option “1” (Preprocessor).

   The CSS Utilities - Preprocessor screen (Figure 3-4) appears, displaying a series of fields for you to complete.

   Note: Before running the preprocessor, you should specify a Hold Queue SYSOUT Class on the CSS Utilities - General Settings screen (Figure 3-2 on page 3-4). To display general settings, select option **SE**.

Figure 3-4. CSS Utilities - Preprocessor Screen

```
------------- COMPUWARE CSS UTILITIES --- Convert Your Compile JCL ---- V8.0.0
COMMAND ===>

Commands: blank (Process JCL)   Browse  Edit  SEtup  C (Create DDIO file)
          LP (LP tutorial)   DD (DDIO tutorial)   SD (SD/DB tutorial)

ISPF Library:
  Project ===>
  Group   ===>          ===>          ===>          ===>
  Type    ===>
  Member  ===>               (Blank for member selection list)

Other Partitioned or Sequential Dataset:

   Dataset Name ===>
  Volume Serial ===>

Language Processor Related Items:
DDIO Listing File ===> LISTFILE
 Options Dataset ===> 'CX.QA.PARMLIB(CXLPCOBB)'
  Enhanced FIND ===> N      (Y/N)        (Xpediter/TSO only)


            Press ENTER to process or enter END command to terminate
```

3. Complete the fields on this screen using the following table as a guide.

Table 3-2. Preprocessor Screen Fields

To:	Do This:
Specify the dataset containing your JCL	Complete the ISPF Library or Other Partitioned or Sequential Dataset fields.
View a list of members in a partitioned dataset	1. Specify a dataset in the ISPF Library or Other Partitioned or Sequential Dataset fields. Ensure that the ISPF Library Member field is blank. 2. Ensure that the Command prompt is blank. 3. Press **Enter**.
Browse a dataset or a particular dataset member	1. Specify a dataset or a fully-qualified member name in the ISPF Library or Other Partitioned or Sequential Dataset fields. 2. Enter **BROWSE** at the Command prompt. 3. Press **Enter**.
Edit a dataset or a particular dataset member	1. Specify a dataset or a fully-qualified member name in the ISPF Library or Other Partitioned or Sequential Dataset fields. 2. Enter **EDIT** at the Command prompt. 3. Press **Enter**.
Create a source listing database file	1. Enter **C** at the command prompt. 2. See “Creating and Formatting Source Listing Files” on page 3-12. 3. Press **PF3** on Create screen to return to this screen.

Table 3-2. Preprocessor Screen Fields

To:	Do This:
Specify the source listing file that will store your source listing report	Enter the name of a new or existing source listing file in the DDIO Listing File field.
Instruct the Language Processor to preprocess the program using the language processor options specified in a particular dataset	Enter the dataset name in the Options Dataset field. **Note:** An Options Dataset overrides any CWPPRMO dataset specification in your JCL.
Edit the CSS Utilities General Settings	Enter **SETUP** at the Command prompt.
Generate records for the XPEDITER/TSO enhanced FIND feature	Enter **Y** in the Enhanced FIND field if you want to use this feature. Enter **N** in the Enhanced FIND field if you do not want to use this feature.
Enter a job statement (if your JCL does not contain a job statement)	1. Enter **SETUP** at the Command prompt 2. On the General Settings Screen that appears, enter a job statement 3. Press **Enter**.

4. After you have finished editing the fields, perform the appropriate action:
   - To convert your JCL to compile with the Compuware Language Processor, ensure that the Command prompt is blank and press **Enter**.
   - To cancel the changes and return to the previous screen, enter **END** at the Command prompt.

Selecting Specific Members for Processing

If you enter a complete library name or a partitioned dataset name on the Preprocessor screen, a list appears containing all members in the specified dataset. Table 3-3 lists the line commands available on this screen.

Table 3-3. Line Commands on Member Selection Screen.

Enter:	To Do This:
B	Browse the member
E	Edit the member
S	Select the member for processing

Members may be sorted using any of the following criteria: Name, Created, Changed, or ID. To sort the member list, enter **SORT** at the command prompt and specify the sort characteristic (for example, SORT CHANGED).

To cancel and return to the previous screen, enter **END** at the command prompt.

Submitting Your JCL for Processing

To submit your JCL for processing from an Edit screen (for example, if you entered **E** on the member selection screen), enter **SUB** at the command prompt.

Postprocessing

Postprocessing performs the following steps:

1. Reads an existing compiler listing and gathers information from the source listing, XREF, data maps, and object code sections of the listing.

2. Gathers this information into a source listing report.
3. Stores the source listing report as a member in an existing source listing DDIO file.

You can run the postprocessor in either of two ways:

- using sample JCL provided with CSS
- using CSS Utilities

Running the Postprocessor Using Sample JCL

To run the postprocessor using sample JCL provided with CSS, see the Language Processor chapter appropriate for your language: **Chapter 9, "COBOL Language Processor"**, **Chapter 10, "PL/I Language Processor"**, **Chapter 11, "Assembler Language Processor"** or **Chapter 12, "C Language Processor"**.

Running the Postprocessor Using CSS Utilities

CSS Utilities can be used to generate or update an existing job stream to postprocess a complete listing and store it in a source listing DDIO or source shared directory database file.

To use CSS Utilities to run the postprocessor, complete the following steps:

1. Select CREATEJCL from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).

   The CSS Utilities - CREATEJCL Menu (Figure 3-3 on page 3-5) is displayed.

2. Select option 2 Postprocessor.

   The CSS Utilities - Postprocessor screen (Figure 3-5) appears, displaying a series of fields for you to complete.

Figure 3-5. CSS Utilities - Postprocessor Screen

```
------------- COMPUWARE CSS UTILITIES --- Load Existing Listing ------- V8.0.0
COMMAND ===>

Commands: blank (Process Listing)   Browse  Edit  SEtup  C (Create DDIO file)
          LP (LP tutorial)   DD (DDIO tutorial)   SD (SD/DB tutorial)


Compiler Listing  ===> SYSPRINT
     Preparation  ===> EDITJCL       Editjcl/Batch
        Language  ===> COBOL         COBOL/PLI/ASSEMBLER



DDIO Listing File  ===> LISTFILE
 Options Dataset  ===>
  Enhanced FIND  ===> N      (Y/N)      (Xpediter only)



      Press ENTER to submit the job if preparation is "Batch", or to
      view the job in an edit-session if preparation is "Editjcl".

      Enter END command to terminate.
```

3. Complete the fields on this screen using the following table as a guide.

Table 3-4. Postprocessor Screen Fields

To:	Do This:
Specify the dataset containing your compiler listing	Enter the Dataset name in the Compiler Listing field.

Table 3-4. Postprocessor Screen Fields

To:	Do This:
View a list of members in a partitioned dataset	1. Enter the dataset name in the Compiler Listing field. 2. Ensure that the Command prompt is blank. 3. Press **Enter**.
Browse a dataset or a particular dataset member	1. Enter the dataset or fully-qualified member name in the Compiler Listing field. 2. Enter **BROWSE** at the Command prompt. 3. Press **Enter**.
Edit a dataset or a particular dataset member	1. Enter a dataset or fully-qualified member name in the Compiler Listing field. 2. Enter **EDIT** at the Command prompt. 3. Press **Enter**.
Create a source listing database file	1. Enter **C** at the command prompt. 2. See "Creating and Formatting Source Listing Files" on page 3-12. 3. Press **PF3** on the Create screen to return to this screen.
Use batch processing to submit your job	Enter **BATCH** in the Preparation field.
Edit the JCL generated during postprocessing before creating a source listing member	Enter **EDITJCL** in the Preparation field.
Specify the language used to create your program	Specify **COBOL**, **PLI**, **ASSEMBLER**, or **C** in the Language field.
Specify the source listing file that will store your source listing report	Enter the name of a new or existing source listing DDIO file in the DDIO Listing File field.
Instruct the Language Processor to postprocess the compiler listing using language processor options specified in a particular dataset	Enter the dataset name in the Options Dataset field.
Generate records for the XPEDITER/TSO enhanced FIND feature	Enter **Y** in the Enhanced FIND field if you want to use this feature. Enter **N** in the Enhanced FIND field if you do not want to use this feature.
Edit the CSS Utilities General Settings	Enter **SETUP** at the Command prompt.
Enter JCL job statement information	Enter a job statement in the Job Statement fields.

4. After you have finished editing the fields, perform the appropriate action:
   - To perform postprocessing, ensure that the Command prompt is blank and press Enter.
   - To cancel the changes and return to the previous screen, enter **END** at the Command prompt.

Selecting Specific Members for Processing

If you enter a PDS name in the Compiler Listing field on the Postprocessor screen, a list appears containing all members in the specified dataset. Table 3-5 lists the line commands available on this screen.

Table 3-5. Line Commands on Member Selection Screen

Enter:	To Do This:
B	Browse the member
E	Edit the member
S	Select the member for processing

Members may be sorted using any of the following criteria: Name, Created, Changed, or ID. To sort the member list, enter SORT at the command prompt and specify the sort characteristic (for example, SORT CHANGED).

To cancel and return to the previous screen, enter **END** at the command prompt.

Submitting Your JCL for Processing

To submit your JCL for processing from an Edit screen (for example, if you entered **Editjcl** on the Postprocessor screen), enter **SUB** at the command prompt.

SOURCE - Working with Source Listing Files

Source listing files are DDIO, shared directory, and database files that contain one or more source listing members. CSS Utilities enables you to perform a number of source listing file functions, including:

- Creating and formatting source listing files
- Reformatting existing source listing files (existing contents are deleted)
- Displaying source listing files
- Copying, moving, importing, and exporting source listing file members
- File manipulation (for example, file deletion, verify shared directory, list attached databases (for shared directory), change auto-delete characteristics on a source database, etc.).

To work with source listing files, select **Source** from the CSS Utilities - Primary Menu (see Figure 3-1 on page 3-2). The CSS Utilities - Source Listing File Facility screen (Figure 3-6) is displayed.

Figure 3-6. CSS Utilities - Source Listing File Facility Screen

```
----------  CSS UTILITIES - Source Listing File Facility  -------- CSS V8.0.0
COMMAND ===>
Commands: SEtup   DDIOCALC (DDIO calculator)
                  DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

Processing Options for S/Member List:        Source Browsing Options:
    Confirm Delete ===> YES (Yes or No)              Unit ===> VIO
  Show print setup ===> YES (Yes or No)           Blocking ===> 10
          Language ===> ENGLISH           (ENGLISH/JAPANESE/USAMIX/USAUC)

SEL Options:  C (Create)  I (Info)  M (Move)  S (Member List)  A (Adv Function)

    SEL     --------------- Source Listing Files -------------------------

     _      ________________________________________________________________
     _  (1)  'CX.DDIO.LISTFILE'
     _  (2)  'CX.SOURCE.SHRDIR'
     _  (3)  'CX.SOURCE.DATABASE'
     _  (4)  'CX.SOURCE.DB1'
     _  (5)  'CX.R070601.DDIO'
     _  (6)
     _  (7)
     _  (8)
```

Setting Source Listing File Options

The CSS Utilities - Source Listing File Facility screen (Figure 3-6) enables you to set a number of source listing file options (described in Table 3-6). These options apply to the member list displayed when the **S (Member List)** option is selected.

For a listing of advanced parameters, see “Setting Advanced Parameters” on page 3-29.

Table 3-6. Source Listing File Options

To:	Do This:
Edit the CSS Utilities General Settings	Enter **SE** at the Command prompt
Invoke the DDIOCALC function.	Enter **DDIOCALC** at the Command prompt
Specify whether a confirmation screen will be displayed before deleting members	Enter one of the following in the Confirm Delete field: **Y:** (Default) Displays a confirmation screen before deleting a member from the source listing file **N:** Deletes a member without displaying a confirmation screen
Specify whether hardcopy options will be displayed before a hardcopy of a source listing member is generated	Enter one of the following in the Show Print Setup field: **Y:** (Default) Displays hardcopy options before creating a hardcopy **N:** Generates a hardcopy without displaying hardcopy options
Specify the language in which the member list will be displayed when **S (DDIO member list)** is selected.	Enter one of the following languages in the Language field: **English:** (Default) Displays the member list in mixed-case English. **Japanese:** Displays the member list in Japanese. Requires a terminal that supports DBCS. **USAMIX:** Displays the member list in mixed-case English. **USAUC:** Displays the member list in upper-case English.

Table 3-6. Source Listing File Options

To:	Do This:
Specify the type of device that will store the temporary dataset	Enter **VIO**, any valid physical address, type, or generic name in the Unit field. **Note:** VIO improves response time.
Specify the size of the temporary dataset	Enter a number of records per block in the Blocking field. **Note:** Entering a larger number improves response time. However, if the Blocking value is too large, it will be reduced to the maximum value permitted for the selected device type.

Creating and Formatting Source Listing Files

On the CSS Utilities - Source Listing File Facility screen (Figure 3-6 on page 3-11), the Source Listing Files section lists the eight source listing files that have been most recently used. To create and format source listing files, complete the following steps.

1. Perform the appropriate step:
   - To reformat an existing source listing file, scan the Source Listing Files List on this screen. If it does not contain the file you want to reformat, enter the file name on the blank line at the top of the list. Enter **C** to the left of the file name
   - To create a new source listing file (and add the file to the list), enter the file name at the entry line under **Source Listing Files**, then enter a **C** to the left of the file name.
2. Press **Enter**.

   If the file name you entered does NOT exist yet, you will first see the following file-type selection panel. You must select which type of file you want to create. Based on the type selected, one of the following “create” screens will then appear:

Figure 3-7. CSS Utilities - Select Desired File Type

```
--------CSS UTILITIES - Select Desired File Type -------CSS V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  File Name entered: 'CX.NEW.FILE'


          1  Shared Directory    - Create a shared directory
                                   NOTE: At least one database file must be
                                   attached to this shared directory for it
                                   to be functional.


          2  Database File       - Create a DDIO/Database that will be
                                   attached to an EXISTING shared directory

          3  DDIO File           - Create a non-database DDIO file


                     Enter END command to terminate
```

The CSS Utilities - Create/Format Source Listing File screen (Figure 3-8) is displayed for existing standard DDIO files, CSS Utilities - Create/Format Shared Directory (Figure 3-9) is displayed for existing shared directory files, and CSS Utilities - Create/Format Source Listing Database (Figure 3-10 on page 3-14) is displayed for existing Source Listing Database files attached to a shared directory.

Figure 3-8. CSS Utilities - Create/Format Source Listing File Screen (Standard DDIO Files)

```
--- COMPUWARE CSS UTILITIES - CREATE/FORMAT Source Listing File ---- V7.9.0
COMMAND ===>

Commands: SEtup    A (Advanced Parms)    D (Delete file)
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

       Dataset Name ===> 'CX.LISTFILE'

        Preparation ===> EDITJCL    (Batch/Editjcl)
    DS Organization ===> VSAM       (Vsam/Sequential)
         Reallocate ===> NO         (Yes/No)

          Model DSN ===>
      Storage Class ===>            (optional)
   Management Class ===>            (optional)
         Data Class ===>            (optional)
             Volume ===>            (optional)
               Unit ===>            (optional)

        Space Units ===> TRACKS     (Blocks/Tracks/Cylinders)
   Primary Quantity ===> 200
  Number of Members ===> 100        (1 to 32767)

           Press ENTER to process  or  enter END command to terminate
```

Figure 3-9. CSS Utilities - Create/Format Source Listing Shared Directory

```
--- COMPUWARE CSS UTILITIES - CREATE/FORMAT Shared Directory ----- CSS V7.9.0
COMMAND ===>
Commands: SEtup    A (Advanced Parms)
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

       Dataset Name ===> 'CX.NEWFILE'

        Preparation ===> EDITJCL    (Batch/Editjcl)
         Reallocate ===> YES        (Yes/No)

          Model DSN ===> 'CX.NEW.LP.SD'
      Storage Class ===>            (optional)
   Management Class ===>            (optional)
         Data Class ===>            (optional)
             Volume ===>            (optional)

        Space Units ===> TRACKS     (Blocks/Tracks/Cylinders)
   Primary Quantity ===> 100
           OR
  Number of Entries ===> 100        (1 to 32767)      (optional)
  (This field overrides above space parms & calculates space needed)

           Press ENTER to process  or  enter END command to terminate
```

Figure 3-10. CSS Utilities - Create Source Listing Database (Attached to a Shared Directory)

```
- COMPUWARE CSS UTILITIES - CREATE/FORMAT Source Database- CSS V7.9.0
COMMAND ===>

Commands: SEtup    A (Advanced Parms)
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

       Dataset Name ===> 'CX.LP.DATABASE'

        Preparation ===> EDITJCL    (Batch/Editjcl)
    DS Organization ===> V          (Vsam/Sequential)
         Reallocate ===> YES        (Yes/No)

          Model DSN ===> 'CX.LP.DATABASE'
      Storage Class ===>            (optional)
   Management Class ===>            (optional)
         Data Class ===>            (optional)
             Volume ===>            (optional)
               Unit ===>            (optional)

        Space Units ===> CYLINDERS  (Blocks/Tracks/Cylinders)
   Primary Quantity ===> 10
  Number of Members ===> 100        (1 to 32767)
   Shared Directory ===> 'CX.LP.SHRDIR'

          Press ENTER to process  or  enter END command to terminate
```

3. Complete the fields on these screens using the following table as a guide.

Table 3-7. Create/Format Source Listing File Fields

To:	Do This:
Edit the CSS Utilities General Settings	Enter **SE** at the Command prompt.
Display advanced parameters	Enter **A** at the Command prompt. For a complete description of the advanced parameters, see Table 3-8 on page 3-18.
Delete the source listing file specified in the Dataset Name field	Enter **D** at the Command prompt. (only on standard DDIO create)
Use batch processing to submit your job	Enter **BATCH** in the Preparation field.
Generate JCL for the Create/Format job, then edit the JCL	Enter **EDITJCL** in the Preparation field.
Specify the file type	Enter **VSAM** or **SEQUENTIAL** in the DS Organization field. **Note:** If a Model DSN is specified (see below), this parameter is optional. If a Model DSN is not specified, you must specify a file type.
Delete and reallocate the specified source listing file	Enter **YES** in the Reallocate field.
Specify a "template" (another source listing file) for allocating and formatting source listing files	**(Optional)** Enter the dataset name in the Model DSN field. **Note:** All explicitly set parameters on this screen, and on the Advanced Parameters screen, override settings in the template file.
Specify a Storage Management Subsystem (SMS) storage class	**(Optional)** Enter an eight-character storage class in the Storage Class field.
Specify an SMS management class	**(Optional)** Enter an eight-character management class in the Management Class field.
Specify an SMS data class	**(Optional)** Enter an eight-character data class in the Data Class field.

Table 3-7. Create/Format Source Listing File Fields

To:	Do This:
Specify the volume serial number of the source listing file	**(Optional)** Enter the volser in the Volume field.
Specify a device or device type	**(Optional)** Enter the device or device type in the Unit field.
Specify a unit of measure for the file size	Enter BLOCKS, TRACKS, or CYLINDERS in the Space Units field. **Note:** If DS Organization = VSAM and a Model DSN is specified, this parameter is optional. If not, this parameter is required.
Specify an amount of primary space for the source listing file	Enter a value in the Primary Quantity field. **Note:** If DS Organization = VSAM and a Model DSN is specified, this parameter is optional. If not, this parameter is required.
(DDIO or Database only) Specify the maximum number of members that the source listing file can contain	Enter a value in the Number of Members field. **Note:** If a Model DSN is specified, this parameter is optional. If not, this parameter is required.
(Shared Directory only) Specify the number of entries, total number of members, to be stored In the shared directory.	**(Optional)** Enter a value in the Number of Entries field. If a value is entered here, it will override the space Parameters entered.
(Source Listing Database Only) Specify the shared directory	Enter the name of the Shared directory to which you want to attach the Database.

4. After you have finished editing the fields, perform the appropriate action:
   - To submit the Create/Format job, ensure that the Command prompt is blank and press **Enter**.
   - To cancel the changes and return to the previous screen, enter **END** at the Command prompt.

Setting Advanced Create/Format Parameters

To edit source listing file Create/Format advanced parameters, complete the following steps:

1. Enter **A** at the command prompt on the CSS Utilities - Create/Format Source Listing File screen (Figure 3-8 on page 3-13).
2. Make sure the DS Organization field has been set to the desired file type.
3. Press **Enter**.

Note: Figure 3-11 displays the Source Listing File Create/Format - Advanced Parameters screen for Sequential Source Listing files. Figure 3-12 displays the Source Listing File Create/Format - Advanced Parameters screen for VSAM files.

Figure 3-11. Source Listing File Create/Format - Advanced Parameters Screen (Sequential)

```
----- SOURCE LISTING FILE CREATE/FORMAT - ADVANCED PARAMETERS ------ V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Prompt this panel upon process ===> N  (Y/N)

  Enter/Modify parameters for Sequential Source Listing file:


                                                     DEFAULTS:
   Blocksize ===>         (2048 to TRKSIZE)            4096



  Groupcount ===>         (1 to 2048)                     4
  Autodelete ===> DUPS    (Dups/Yes/No/Staged)         Dups
     Extents ===>         (1 to 460)                     64


  Note: Once any of the above parameters are modified, the new values will
        remain in effect for future formatting until they are modified again.

         Press ENTER to process  or  enter END command to terminate
```

Figure 3-12. Source Listing File Create/Format - Advanced Parameters Screen (VSAM)

```
------ Source Listing File CREATE/FORMAT - ADVANCED PARAMETERS ------ V8.0.0
 COMMAND ===>
 Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

   Prompt this panel upon process ===> N  (Y/N)

   Enter/Modify parameters for VSAM Source Listing file:

                                                         DEFAULTS:
           CISIZE ===>                                       4096
        Secondary ===> YES     (Yes or No)                    NO
     Sec.Quantity ===> 2       (Uses primary space units)
     VSAM Catalog ===>

       Groupcount ===>         (1 to 2048)                     4
       Autodelete ===> YES     (Dups/Yes/No/Staged)         Dups
          Extents ===>         (1 to 460)                     64


  Note: Once any of the above parameters are modified, the new values will
        remain in effect for future formatting until they are modified again.

         Press ENTER to process  or  enter END command to terminate
```

Figure 3-13. Shared Directory Create/Format - Advanced Parameters Screen

```
----- Shared Directory  CREATE/FORMAT - ADVANCED PARAMETERS ---- CSS V7.9.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Prompt this panel upon process ===> N  (Y/N)

  Enter/Modify parameters for VSAM Shared Directory:

                                                        DEFAULTS:
         CISIZE ===> 7000                                   4096
      Secondary ===> NO      (Yes or No)                     NO

    VSAM Catalog ===>





 Note: Once any of the above parameters are modified, the new values will
       remain in effect for future formatting until they are modified again.

         Press ENTER to process or enter  END command to terminate
```

Figure 3-14. Source Listing Source Listing Database Create/Format - Advanced Parameters Screen (VSAM)

```
------- Source Database  CREATE/FORMAT - ADVANCED PARAMETERS ---- CSS V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Prompt this panel upon process ===> N  (Y/N)

  Enter/Modify parameters for VSAM Source Database:


                                                        DEFAULTS:
         CISIZE ===> 8192                                   4096
      Secondary ===>         (Yes or No)                     NO

    VSAM Catalog ===>

     Groupcount ===> 8       (1 to 2048)                      4
     Autodelete ===> DUPS    (Dups/Yes/No/Staged)          Dups
        Extents ===>         (1 to 460)                      64


 Note: Once any of the above parameters are modified, the new values will
       remain in effect for future formatting until they are modified again.
```

Figure 3-15. Source Listing Source Listing Database Create/Format - Advanced Parameters Screen (Sequential)

```
------- Source Database  CREATE/FORMAT - ADVANCED PARAMETERS --- CSS V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Prompt this panel upon process ===> N  (Y/N)

  Enter/Modify parameters for Sequential Source Database:


                                                   DEFAULTS:
   Blocksize ===>         (2048 to TRKSIZE)           4096



  Groupcount ===> 8       (1 to 2048)                    4
  Autodelete ===> DUPS    (Dups/Yes/No/Staged)        Dups
     Extents ===>         (1 to 460)                    64



  Note: Once any of the above parameters are modified, the new values will
        remain in effect for future formatting until they are modified again.
```

4. Complete the fields on this screen, as necessary, using the following table as a guide.

Table 3-8. Create/Format Source Listing File Advanced Parameters

To:	Do This:
Specify whether advanced parameters will be displayed each time a Create/Format Source Listing File command is processed	Enter one of the following in the Prompt This Panel Upon Process field: **Y:** Displays advanced parameters before processing Create/Format commands. **N:** Processes Create/Format commands without displaying advanced parameters.
(Sequential files only) Specify the source listing file blocksize	Enter an integer between 2048 and 32760 in the Blocksize field. • If you do not complete this field, the default blocksize of 4096 will be used. • The blocksize is always ignored during the file allocation phase.
(VSAM files only) Specify the control interval size	Enter an integer in the CISIZE field. If you do not complete this field, the default value of 4096 will be used.
(VSAM files only) Specify whether VSAM source listing files are allocated with secondary extents	Enter one of the following in the Secondary field: **Yes:** VSAM source listing files will be allocated with secondary extents. **No:** (Default) VSAM source listing files will not be allocated with secondary extents.
(VSAM files only) Specify the amount of secondary space for the specified source listing file (used when the primary space has been consumed)	Enter a number of blocks, tracks, or cylinders in the Sec. Quantity field. **Note:** The unit of measure (blocks, tracks, or cylinders) is determined by the Space Units setting on the CSS Utilities - Create/Format Source Listing File screen (Figure 3-8 on page 3-13).
(VSAM files only) Specify the VSAM catalog in which the source listing file will be created	**(Optional)** Enter a catalog name in the VSAM Catalog field.

Table 3-8. Create/Format Source Listing File Advanced Parameters

To:	Do This:
Specify the number of blocks allocated to a source listing file when additional space is needed	Enter a number between 1 and 2048 in the Groupcount field. **Note:** Compuware recommends setting the Groupcount to **8**. The default setting is **4**.
Specify whether the oldest member of a source listing file will be deleted when the file becomes full	Enter one of the following settings in the Autodelete field: **Dups:** (Default) Unlocks automatically locked members with the same name, then locks the most recent member. **Yes:** Automatically deletes one or more of the oldest unlocked members to allow new members to be written to the file. **No:** You must manually delete members when the DDIO file becomes full. **Staged:** Similar to DUPS. For a complete description of these settings, see "Command Parameters for DDIO Files" on page 4-6 if it is a source listing DDIO file; and "Command Parameters for Shared Directories and Source Listing Databases" on page 7-6, if it is a source listing database.
Specify the maximum number of extents allocated to each source listing member	Enter a number between 1 and 460 in the Extents field. **Note:** If you do not complete this field, the default value of 64 will be used.

5. After you have finished editing the fields, perform the appropriate action:
   - To apply the advanced parameter settings and return to the previous screen, ensure that the Command prompt is blank and press **Enter**.
   - To cancel the changes and return to the previous screen, enter **END** at the Command prompt.

Deleting Source Listing Files

To delete a source listing file, complete the following steps.

1. Ensure that the CSS Utilities - Source Listing File Facility screen (Figure 3-6 on page 3-11) is displayed.

   You can display this screen by selecting option 2 (Source) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).

2. Scan the Source Listing Files list on this screen. If it does not contain the file that you want to delete, enter the file name on the blank line at the top of the list.

For standard DDIO files only - Steps 3 - 7:

3. Enter **C** to the left of the file name.
4. Press **Enter**.

   The CSS Utilities - Create/Format Source Listing File screen (Figure 3-8 on page 3-13) is displayed.

5. Ensure that the correct source listing file is specified in the Dataset Name field.
6. Enter **D** at the command prompt.
7. Press **Enter**.

For all types of source listing files - Steps 8 - 12:

8. Or, optionally, (instead of steps 3 through 7) enter **A** to the left of the file name

9. Press **Enter**.
10. The CSS Utilities Advanced Functions panel will be displayed.
11. Select the “Delete” option from the menu, press **Enter**.
12. Follow instructions on the particular “delete” function panel that is displayed (different by file type). See “Setting Advanced Parameters” on page 3-29.

Displaying File Information

To display information about a particular source listing file, complete the following steps:

1. Ensure that the CSS Utilities - Source Listing File Facility screen (Figure 3-6 on page 3-11) is displayed.

   You can display this screen by selecting option 2 (Source) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).
2. Scan the DDIO Files list on this screen. If it does not contain the file for which you want to view information, enter the file name on the blank line at the top of the list.
3. Enter **I** to the left of the file name.
4. Press **Enter**.

A report (Figure 3-16) is displayed containing current parameter settings and file statistics. To scroll through the report, use the **PF7** and **PF8** keys.

Figure 3-16. Source Listing File Information Report

```
COMPUWARE/VF ----- 'CX.LISTFILE' --------------------------- Row 1 to 21 of 41
COMMAND INPUT ===>                                            SCROLL ===> PAGE
                      DIRECTORY INFORMATION

DSN ------------------------------ CX.LISTFILE
 VOLSER --------------------------- PRD921
 AM(ACCESS METHOD) --------------- BSAM
 BLKSIZE ------------------------- 8185
 TYPE ---------------------------- SOURCE
 REPTCOUNT ----------------------- 100
 GROUPCOUNT ---------------------- 4
 AUTODELETE ---------------------- DUPS
 INDEX --------------------------- EXT
 EXTENTS ------------------------- 256
 OTHER INFORMATION
   BLOCKS IN FILE ---------------- 360
   ALLOCATION GROUPS ------------- 87
   USED ALLOCATION GROUPS -------- 85(98%)
   BLOCKS/TRACK ------------------ 6
   FILE FORMATTED BY DDIO RELEASE- 7.4.0
   REQUIRED MINIMUM DDIO RELEASE - 5.6.0
   CURRENT REPORT NUMBER --------- 45
   DIRECTORY ENTRIES/BLOCK ------- 13
   DIRECTORY BLOCKS -------------- 8
```

Note: To see a complete report sample and a description of the fields on the report, refer to “Source Listing File DIRX Report” on page 2-17.

Figure 3-17. Source Listing Database Information Screen

```
COMPUWARE/VF ----- 'CX.LP.DATABASE' ------------------------ Row 1 to 20 of 44
COMMAND INPUT ===>                                            SCROLL ===> CSR
                        DIRECTORY INFORMATION
 DATABASE IS CONNECTED
 CATALOG DSN: CX.LP.CATALOG4
      VOLSER: PRD988

 DSN ----------------------------- CX.LP.DATABASE
 VOLSER -------------------------- PRD925
 AM(ACCESS METHOD) --------------- BSAM
 BLKSIZE ------------------------- 12288
 TYPE ---------------------------- SOURCE
 REPTCOUNT ----------------------- 40
 GROUPCOUNT ---------------------- 1
 AUTODELETE ---------------------- YES
 INDEX --------------------------- EXT
 EXTENTS ------------------------- 256
 OTHER INFORMATION
   BLOCKS IN FILE ---------------- 200
   ALLOCATION GROUPS ------------- 196
   USED ALLOCATION GROUPS -------- 166(85%)
   BLOCKS/TRACK ------------------ 4
   FILE FORMATTED BY DDIO RELEASE- 7.7.0
```

Note: To see a complete report sample and a description of the fields on the report, refer to "Source Listing Database DIRX Report" on page 2-49.

Figure 3-18. Source Listing Shared Directory Information Screen

```
COMPUWARE/VF ----- 'CX.LP.SHRDIR' -------------------------- Row 1 to 20 of 110
COMMAND INPUT ===>                                            SCROLL ===> CSR
                        DIRECTORY INFORMATION
 DIRX FOR SHARED DIRECTORY: CX.LP.SHRDIR
                    VOLSER: PRD988


 SHARED DIRECTORY INFORMATION:
 AM ------------------------------ VSAM
 BLKSIZE ------------------------- 7161
 OTHER INFORMATION
   DIRECTORY FORMATTED BY DDIO --- 7.8.0
   REQUIRES MINIMUM DDIO RELEASE - 7.0.0
   TOTAL BLOCKS IN SHARED DIR ---- 105
   USED BLOCKS ------------------- 27(26%)
   NUMBER OF EXTENTS ------------- 1


   DIRECTORY CLASS: DATABASE
   CLASS INSTANCE NAME ----------- DATABASE
   CLASS ANCHOR RBN -------------- 12
   NUMBER OF ENTRIES ------------- 0
   CURRENT PUT NUMBER ------------ 1
```

Note: To see a complete report sample and a description of the fields on the report, refer to "Source Listing Shared Directory (SRCEDIR) DIRX Report" on page 2-56.

Working with Source Listing File Members

CSS Utilities enables you to work with source listing file members in the following ways:

- Display members
- Copy members
- Move members
- Export members

- Import members

These operations are described in the following sections.

Displaying Members

To display the members in a particular source listing file, complete the following steps:

1. Ensure that the CSS Utilities - Source Listing File Facility screen (Figure 3-6 on page 3-11) is displayed.

   You can display this screen by selecting option 2 (Source) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).

2. Scan the Source Listing Files list on this screen. If it does not contain the desired file, enter the file name on the blank line at the top of the list.

3. Enter **S** to the left of the file name.

4. Press **Enter**.

   A member list (Figure 3-19) is displayed. To scroll through the list, use the **PF7** and **PF8** keys. To display a list of the line commands available on this screen, press **PF1**.

Figure 3-19. Source Listing File Member List - DDIO Format

```
COMPUWARE/VF -- Source Listing File Member List ---          Row 1 to 15 of 40
COMMAND INPUT ===>                                           SCROLL ===> PAGE
Source DDIO File: CX.LISTFILE


SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
______ COBABEND  (L)        47    25 JUN 2002  11:02:41  04   COBIIR4        32
______ COBABEND             45    13 MAY 2001  11:03:11  04   COBOL/390M     64
______ COBABEND             46    06 MAY 2001  14:21:34  04   COBIIR4        32
______ COBABEND  (M)        43    06 MAY 2001  14:21:34  00   COBIIR4        32
______ COBABEND             30    28 JAN 2000  16:21:32  00   COBOL/390      32
______ FEA02P2   (L)        29    15 JAN 2000  09.38.39  04   PL/I V1R5     128
______ FEA02P2              28    15 JAN 2000  09.18.11  04   PL/I V1R5     128
______ FEA02P2              27    15 JAN 2000  09.05.47  04   PL/I V1R5     128
______ FEA02P2              26    13 JAN 2000  15.14.37  04   PL/I V1R5     128
______ FEA02P2              25    13 JAN 2000  14.28.19  04   PL/I V1R5     128
______ FEA02P2              24    13 JAN 2000  14.23.59  04   PL/I V1R5     128
______ PLIBASD              21    13 JAN 2000  10.48.37  00   PL/I V2       128
______ PLIBASD   (L)        23    13 JAN 2000  01.05.45  04   PL/I V2        32
______ PLIBASD              22    13 JAN 2000  01.04.40  00   PL/I V2        32
______ COBABEND             20    13 JAN 2000  15.56.17  00   COBOL/390      32
```

Figure 3-20 shows the Source Listing Shared Directory Member List screen with “Display Long PROGRAM Names” set to NO. A LONGNAME will be indicated by an ‘*’ in the eighth position of the name. A mixed case name is shown as stored.

Note: Selection **I-Info** is valid only for Source Listing Database and Shared Directory.

Figure 3-20. Source Listing Shared Directory Member List Screen (Not LONGNAME)

```
COMPUWARE/VF -- Source Listing Shared Directory Member List -- Row 1 to 7 of 7
COMMAND INPUT ===>                                          SCROLL ===> CSR
Shared Directory: CX.LP.SHRDIR

Display Long PROGRAM Names ==> NO  (yes or no) Number of lines ==>  1  (1 TO 4)

SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock   I - Info
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
______ AVery     (L)       215    07 SEP 2001  08.55.34  00   C/390 V2R6   1032
______ ASMTEST   (L)       199    21 APR 2000  15.42     00   ASMH           60
______ COBABND6  (M)       192    18 APR 2000  06:47:02  00   COBIIR4        48
______ AVeryLa*  (L)       191    14 APR 2000  11:16:29  00   C/ESA V3R2    816
______ COBABND1  (M)       189    12 APR 2000  12:38:15  04   COBIIR4        36
______ COBABEND  (M)       186    04 APR 2000  17:11:10  04   COBIIR4        36
______ COBABND1              7    04 APR 2000  17:11:31  04   COBIIR4        36
****************************** Bottom of data ********************************
```

Figure 3-21 shows the a Source Listing Shared Directory Member List with “Display Long PROGRAM Names” set to YES. A mixed case name is shown as stored.

Figure 3-21. Source Listing Shared Directory Member List Screen (LONGNAME)

```
COMPUWARE/VF -- Source Listing Shared Directory Member List -- Row 1 to 5 of 7
COMMAND INPUT ===>                                          SCROLL ===> CSR
Shared Directory: CX.LP.SHRDIR

Display Long PROGRAM Names ==> YES (yes or no) Number of lines ==>  1  (1 TO 4)

SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock   I - Info
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
______ AVery     (L)       215    07 SEP 2001  08.55.34  00   C/390 V2R6   1032
AVery
SOURCE DATABASE: CX.LP.DATABASE
______ ASMTEST   (L)       199    21 APR 2000  15.42     00   ASMH           60
ASMTEST
SOURCE DATABASE: CX.LP.DATABASE
______ COBABND6  (M)       192    18 APR 2000  06:47:02  00   COBIIR4        48
COBABND6
SOURCE DATABASE: CX.LP.DATABASE
______ AVeryLa*  (L)       191    14 APR 2000  11:16:29  00   C/ESA V3R2    816
AVeryLargeNameofatLeast30Characters_20
SOURCE DATABASE: CX.LP.DATABASE
______ COBABND1  (M)       189    12 APR 2000  12:38:15  04   COBIIR4        36
COBABND1
SOURCE DATABASE: CX.LP.DATABASE
```

Figure 3-22 shows the Source Listing Database Member List screen with “Display Long PROGRAM Names” set to NO. A LONGNAME will be indicated by an ‘*’ in the eighth position of the name. A mixed case name is shown as stored in the Source Listing Shared Directory.

Figure 3-22. Source Listing Database Member List Screen (Not LONGNAME)

```
COMPUWARE/VF -- Source Listing Database Member List -----     Row 1 to 9 of 9
COMMAND INPUT ===>                                            SCROLL ===> CSR
Source Database:  CX.LP.DATABASE
ATTACHED TO:      CX.LP.SHRDIR
Display Long PROGRAM Names ==> NO  (yes or no) Number of lines ==>  1  (1 TO 4)

SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock   I - Info
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
______ ASMTEST             415    26 OCT 2001  15.12     00   ASMH           56
______ COBDEMO1            405    03 OCT 2001  15:41:41  04   COBIIR4        98
______ COBDEMO1            404    27 SEP 2001  13:23:08  04   COBIIR4        98
______ COBDEMO1            403    20 SEP 2001  09:52:21  04   COBIIR4        98
______ COBDEMO1  (L)       402    10 SEP 2001  16:03:47  04   COBIIR4        98
______ ASMTEST   (L)       400    10 SEP 2001  15.55     00   ASMH           56
______ ASMTEST             398    10 SEP 2001  15.08     00   ASMH           56
______ AVeryBi*  (M)       397    06 SEP 2001  15.16.50  00   C/390 V2R6   1035
______ AVeryBi*  (M)       375    21 FEB 2000  13.06.52  00   C/ESA V3R2    826
****************************** Bottom of data ******************************
```

Figure 3-23 shows the Source Listing Database Member List screen with “Display Long PROGRAM Names” set to YES. A mixed case name is shown as stored in the Source Listing Shared Directory.

Figure 3-23. Source Listing Database Member List Screen (LONGNAME)

```
COMPUWARE/VF -- Source Listing  Database Member List -----    Row 5 to 9 of 9
COMMAND INPUT ===>                                            SCROLL ===> CSR
Source Database:  CX.LP.DATABASE
ATTACHED TO:      CX.LP.SHRDIR
Display Long PROGRAM Names ==> YES (yes or no) Number of lines ==>  1  (1 TO 4)

SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock   I - Info
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
______ COBDEMO1  (L)       402    10 SEP 2001  16:03:47  04   COBIIR4        98
COBDEMO1
______ ASMTEST   (L)       400    10 SEP 2001  15.55     00   ASMH           56
ASMTEST
______ ASMTEST             398    10 SEP 2001  15.08     00   ASMH           56
ASMTEST
______ AVeryBi*  (M)       397    06 SEP 2001  15.16.50  00   C/390 V2R6   1035
AVeryBigLongNameofatLeast30Characters_20
______ AVeryBi*  (M)       375    21 FEB 2000  13.06.52  00   C/ESA V3R2    826
AVeryBigLargeNameofatLeast30Characters_20
****************************** Bottom of data ******************************
```

For more information and explanations of the fields, see “Source Listing Dataset Directory” on page 8-12

Copying Members

To copy members from one source listing file to another, complete the following steps:

1. Ensure that the CSS Utilities - Source Listing File Facility screen (Figure 3-6 on page 3-11) is displayed.

   You can display this screen by selecting option 2 (Source) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).

2. Scan the Source Listing Files list on this screen. If it does not contain the file from which you want to copy members, enter the file name on the blank line at the top of the list.

3. Enter **M** to the left of the file name.

4. Press **Enter**. The CSS Utilities - Source Listing File Move Facility screen (Figure 3-24) is displayed.

Figure 3-24. CSS Utilities - Source Listing File Move Facility Screen

```
--- COMPUWARE CSS UTILITIES - Source Listing File Move Facility - V8.0.0
COMMAND ===>

Commands: blank (Process)   SEtup   L (List of members)   A (Advanced parms)
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


   Preparation ===> EDITJCL         (Batch/Editjcl)

        Option ===> COPY            (Copy/Move/Export/Import)

     From File ===> 'CX.LISTFILE'
       To File ===>


   Member Name ===>                 (Optional, specify prefix to be used for
                                     select - can be a wildcardor entire name)

 Member Number ===>                 (Optional, specify a member number to be
                                     used - will override "Member Name" field)


           Press ENTER to process  or  enter END command to terminate
```

5. Complete the fields on this screen using the following table as a guide.

Note: This table describes the COPY, MOVE, EXPORT, and IMPORT commands. You will be using the COPY command, so specify COPY in the Option field.

Table 3-9. Source Listing File Move Facility Fields

To:	Do This:
Use batch processing to submit your job	Enter **BATCH** in the Preparation field.
Edit the JCL generated for the COPY, MOVE, EXPORT, or IMPORT before submitting the JCL for processing	Enter **EDITJCL** in the Preparation field.
Specify the command type (COPY, MOVE, EXPORT, or IMPORT)	Enter the command at the Option prompt.
Specify the source listing file **from which** members will be copied, moved, exported, or the sequential file **from which** members will be imported.	Enter a dataset name in the From File field.
Specify the source listing file **to which** members will be copied, moved, or imported	Enter a dataset name in the To File field. If this dataset does not exist, you will be prompted to enter the **C** command to display the CSS Utilities - Create/Format Source Listing File screen (Figure 3-8 on page 3-13 if From File is a standard DDIO, or Figure 3-25 on page 3-28 if From File is a shared directory or database).
Specify the sequential file **to which** members will be exported	Enter a dataset name in the To File field. If this dataset does not exist, it will be created as part of the generated EXPORT job in the JCL.
Perform commands (COPY, MOVE, IMPORT, EXPORT, and LIST) on members matching a particular prefix	Optional: Enter the prefix in the Member Name field. Prefix can be a wild card (followed by an asterisk), or the entire name. **Note:** If a prefix or name is not specified, commands will be performed on all members in the specified file.
Specify a particular source member for processing by member number	Enter the member number in the Member Number field.

Table 3-9. Source Listing File Move Facility Fields

To:	Do This:
View and select members in the **From File** dataset	Enter **L** at the Command prompt. **Note:** See "Selecting Specific Members for Processing" on page 3-7 for step-by-step instructions.
Edit advanced parameters	Enter **A** at the Command prompt. For a complete description of the advanced parameters, see "Setting Advanced Parameters" on page 3-29.
Edit the CSS Utilities - General Settings	Enter **SETUP** at the Command prompt.

6. After you have finished editing the fields, perform the appropriate action:
   - To submit the COPY, MOVE, IMPORT, or EXPORT, ensure that the Command prompt is blank and press **Enter**.
   - To cancel the changes and return to the previous screen, enter **END** at the Command prompt.

Moving Members

To move members from one source listing file to another, complete the following steps:

1. Ensure that the CSS Utilities - Source Listing File Facility screen (Figure 3-6 on page 3-11) is displayed.

   You can display this screen by selecting option 2 (Source) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).
2. Scan the Source Listing Files list on this screen. If it does not contain the file from which you want to move members, enter the file name on the blank line at the top of the list.
3. Enter **M** to the left of the file name.
4. Press **Enter**.

   The CSS Utilities - Source Listing File Move Facility screen (Figure 3-24 on page 3-25) is displayed.
5. Complete the fields on this screen using Table 3-9 on page 3-25 as a guide.
   - Specify MOVE in the Option field.
   - Enter a dataset name in the To File field.

Exporting Members

To export members from a source listing file to a sequential dataset, complete the following steps:

1. Ensure that the CSS Utilities - Source Listing File Facility screen (Figure 3-6 on page 3-11) is displayed.

   You can display this screen by selecting option 2 (Source) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).
2. Scan the Source Listing Files list on this screen. If it does not contain the file from which you want to export members, enter the file name on the blank line at the top of the list.
3. Enter **M** to the left of the file name.
4. Press **Enter**.

The CSS Utilities - Source Listing File Move Facility screen (Figure 3-24 on page 3-25) is displayed.

5. Complete the fields on this screen using Table 3-9 on page 3-25 as a guide.
   - Specify EXPORT in the Option field.
   - Enter the name of a sequential dataset in the To File field. If this dataset does not exist, it will be created as part of the generated EXPORT job in the JCL.

Importing Members

To import members from a sequential dataset into a source listing file, complete the following steps:

1. Ensure that the CSS Utilities - Source Listing File Facility screen (Figure 3-6 on page 3-11) is displayed.

   You can display this screen by selecting option 2 (Source) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).
2. Enter the name of the file to import members **from** on the blank line at the top of the list.
3. Enter **M** to the left of the file name.
4. Press **Enter**.

   The CSS Utilities - Source Listing File Move Facility screen (Figure 3-24 on page 3-25) is displayed.
5. Complete the fields on this screen as shown in Table 3-9 on page 3-25.
   - Specify IMPORT in the Option field.
   - Enter the name of a source listing file DDIO dataset in the To File field.

Specifying "C" to Create the "TO" File for Shared Directory or Database Files

If you are prompted to enter the "C" command to create the file name specified on the "TO file" field, and if the "FROM file" is either a shared directory or a database file, this file type selection panel will be displayed. You may create either a database that will be attached to an existing shared directory, or you may create a new shared directory and then also must create a database to be attached to it. For Source Listings, see Figure 3-9 and Figure 3-10 on page 3-14.

Figure 3-25. Select Desired File Type Panel

```
--------CSS UTILITIES - Select Desired File Type -------CSS V7.9.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)


  File Name entered: 'CX.NEW.LP.SD'


          1  Shared Directory    - Create a shared directory
                                   NOTE: You must also next create at least
                                         one database file that will be
                                         attached to this shared directory.


          2  Database File       - Create a DDIO/Database that will be
                                   attached to an EXISTING shared directory.


                     Enter END command to terminate
```

Selecting Specific Members for Processing

To copy, move, or import specific members within a dataset, complete the following steps:

1. Enter **L** at the command prompt on the CSS Utilities - Source Listing File Move Facility screen (Figure 3-24 on page 3-25).

   A list appears containing all members in the **From File** dataset (Figure 3-26).

Figure 3-26. Selecting Members for COPY, MOVE, or IMPORT

```
COMPUWARE/VF ----- 'CX.LISTFILE' ---------------- V8.0.0 --- Row 1 to 18 of 39
COMMAND INPUT ===>                                          SCROLL ===> PAGE

       Line Commands:  S (Select for processing)
                     SOURCE LISTING DATASET DIRECTORY
   PROGRAM    LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE    SIZE(K)
  PLIBASD   (M)        43    01 APR 2002  14:10:53  00   PL/I V2        32
  PLITEST   (L)        42    31 MAR 2002  10:32:00  00   PL/I V2        32
  PLIBASD              41    31 MAR 2002  09:06:58  04   PL/I V2        32
  PLIBASD              40    31 MAR 2002  09:05:25  00   PL/I V2        32
  PLIBASD              39    30 MAR 2002  16:25:05  00   PL/I V2        32
  PLIBASD              38    30 MAR 2002  16:14:51  00   PL/I V2        32
  COBABEND  (L)        37    28 JAN 2002  16:21:32  00   COBOL/390      32
  FEA02P2   (L)        35    15 JAN 2002  09.38.39  04   PL/I V1R5     128
  FEA02P2              34    15 JAN 2002  09.18.11  04   PL/I V1R5     128
  FEA02P2              33    15 JAN 2002  09.05.47  04   PL/I V1R5     128
  FEA02P2              32    13 JAN 2002  15.14.37  04   PL/I V1R5     128
  FEA02P2   *I         31    13 JAN 2002  14.49.14  00   PL/I V1R5      32
  FEA02P2   *I         30    13 JAN 2002  14.33.49  00   PL/I V1R5      32
  FEA02P2              29    13 JAN 2002  14.28.19  04   PL/I V1R5     128
  FEA02P2              28    13 JAN 2002  14.23.59  04   PL/I V1R5     128
  FEA02P2   *I         27    13 JAN 2002  11.40.30  00   PL/I V1R5      32
  FEA02P2   *I         26    13 JAN 2002  11.34.09  00   PL/I V1R5      32
  FEA02P2   *I         25    13 JAN 2002  11.22.45  00   PL/I V1R5      32
```

2. Select members by entering an **S** to the left of the member names.

   Selected members are marked with an asterisk and the word **SELECTED**.

3. Press **F3** to return to the CSS Utilities - Source Listing File Move Facility screen (Figure 3-24 on page 3-25).

4. Ensure that the command prompt is blank, then press **Enter**.

Setting Advanced Parameters

To set source listing file advanced parameters, complete the following steps:

1. Ensure that the CSS Utilities - Source Listing File Move Facility screen (Figure 3-24 on page 3-25) is displayed.
2. Enter **A** at the command prompt.

   The Source Listing File Move Function - Advanced Parameters screen (Figure 3-27) is displayed.

Figure 3-27. Source Listing File Move Function - Advanced Parameters

```
----- Source Listing File Move Function - ADVANCED PARAMETERS ---- V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Prompt this panel upon process ===> N  (Y/N)

  Enter/Modify parameters for COPY/MOVE/EXPORT/IMPORT:


                                                        DEFAULTS:
        Age old ===>             nnn
      Age Units ===>             (DAYS/MONTHS/YEARS)    DAYS


       CONTINUE ===>             (YES/NO)               YES
         DELETE ===>             (NO/YES)               NO
           EXEC ===>             (SUMMARY/CHECK/RUN)    SUMMARY
          LABEL ===>             (1 to nn)              1


  Note: Once any of the above parameters are modified, the new values will
        remain in effect for future formatting until they are modified again.

         Press ENTER to process  or  enter END command to terminate
```

3. Edit the parameters as needed, using the following table as a guide.

Table 3-10. Source Listing File Move Function Advanced Parameters

To:	Do This:
Specify whether advanced parameters will be displayed each time a COPY, MOVE, EXPORT, IMPORT, or LIST command is processed	Enter one of the following in the Prompt This Panel Upon Process field: **Y:** Displays advanced parameters before processing COPY, MOVE, EXPORT, IMPORT, and LIST commands. **N:** Processes these commands without displaying advanced parameters
Restrict the COPY, MOVE, EXPORT, and IMPORT commands to members older than **nnn** months, days, or years	1. Specify a number in the Age Old field. 2. Specify a unit (**months**, **days**, or **years**) in the Age Units field.
Specify whether COPY, MOVE, EXPORT, and IMPORT commands will continue if a recoverable error is detected	Enter one of the following in the CONTINUE field: **Y:** (Default) Command continues, if possible, when an error is detected. **N:** Command stops when an error is detected.
Specify whether members will be deleted from the original file (the **From File**) after they are successfully exported	Enter one of the following in the DELETE field: **Y:** Deletes members after they are exported. **N:** (Default) Does not delete members after they are exported.

Table 3-10. Source Listing File Move Function Advanced Parameters

To:	Do This:
Control how the COPY, MOVE, EXPORT, and IMPORT commands are processed *(Not present for Database and/or Shared Directory files)*	Enter one of the following in the EXEC field: **SUMMARY:** (Default) Transfers data and produces a summary report. The summary includes a list of transferred members and a list of any members deleted from the **From File** during the transfer. **CHECK:** Produces a detailed report without moving any data. This report includes a list of transferred members, a list of any members deleted from the **From File** during the transfer, and a list of attributes from the **To** and **From** files. It also includes a list of any members unlocked or deleted during MOVE, COPY, or IMPORT commands. **RUN:** Transfers data and produces a detailed report. See CHECK (above) for a description of the report contents.
Reference multiple export files using a single ddname — create export files on a single tape	Enter the file number of a sequential dataset in the LABEL field. This parameter will be used with the IMPORT and EXPORT commands. You can include the LABEL parameter in your JCL. However, a LABEL parameter specified on this panel overwrites any LABEL setting in your JCL.

4. After you have finished editing the parameters, perform the appropriate action:
   - To apply the advanced parameter settings and return to the previous screen, press **Enter**.
   - To cancel the changes and return to the previous screen, enter **END** at the command prompt.

Advanced Functions - Source

Working with Source Listing Files (File Manipulation)

Figure 3-28. Source DDIO File - Source Listing Advanced Functions - Main Panel

```
------ Source DDIO File - Advanced Functions ----- CSS V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

     Source File: 'CX.LISTFILE'
          Volser: PRD932


          1  Delete       - Delete the source DDIO file




               Enter END command to terminate
```

Select “1” to delete the source DDIO file. The CSS Utilities - Confirm Delete screen appears (see Figure 3-29).

Figure 3-29. Source DDIO File - CSS Utilities CONFIRM DELETE

```
------------------  CSS UTILITIES CONFIRM DELETE   -------------------- V8.0.0
COMMAND ===>

Dataset Name: CX.LISTFILE


Press ENTER to confirm delete process or enter END to cancel this request.
```

Note: This function performs an immediate dynamic file delete, just like using the original "D" function on the create panel.

Working with Source Shared Directory Files (File Manipulation)

Figure 3-30. Source Listing Shared Directory - Advanced Functions - Main Panel

```
--------- Shared Directory  -  ADVANCED FUNCTIONS  --------- CSS V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

Shared Directory: 'CX.LP.SHRDIR'
          Volser: PRD988

          1  LISTDB       - List all attached databases
          2  EXPAND       - Expand size of the shared directory
          3  DELETE       - Delete the shared directory
          4  VERIFY       - Verify the integrity of the shared directory




            Press ENTER to process or enter END command to terminate
```

Select "1" (LISTDB) to list all attached databases. The Shared Directory - Attached Database Report screen appears (see Figure 3-31).

Figure 3-31. Shared Directory - Attached Database Report

```
COMPUWARE/VF --- Shared Directory - Attached Database Report - Row 1 to 3 of 3
COMMAND INPUT ===>                                          SCROLL ===> CSR
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

Shared Directory: 'CX.LP.SHRDIR'
          Volser: PRD988
 I-info  C-change  T-detach  D-delete  R-scratch  S-list member  M-move member

SEL  --------------- Database List --------------    Volser
  _   CX.NEW.LPDDIO2
  _   CX.TEST.LPDDIO3
  _   CX.TEST.LPDDIO5
****************************** Bottom of data ********************************
```

- Select **I** to display the database "display file information" screen (Figure 3-17 on page 3-21).
- Select **C** to display the database "advanced functions - change" screen (Figure 3-39 on page 3-35).
- Select **T** to display the database "advanced functions - detach" screen (Figure 3-38 on page 3-35).
- Select **D** to display the database "advanced functions - delete" screen (Figure 3-40 on page 3-36).

- Select **R** to display the “CSS Utilities - SCRATCH Database” screen (Figure 3-32).
- Select **S** to display the database “member list” screen (Figure 3-22 and Figure 3-23 on page 3-24).
- Select **M** to display the database “move member” screen (Figure 3-24 on page 3-25).

Note: Pressing **PF3** from any of these screens will return you to the Shared Directory - Attached Database Report screen (Figure 3-31).

Figure 3-32. CSS Utilities - SCRATCH Database

```
----------  COMPUWARE CSS UTILITIES - SCRATCH Database ----------  CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


     Database: 'CX.LP.DATABASE'                                Volser: PRD988

  Attached to: 'CX.LP.SHRDIR'                                  Volser:

          Preparation ===> EDITJCL         (Batch/Editjcl)





           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated to perform the “scratch” function and remove the specified database from this shared directory. This should only be done when the database has been destroyed.

Note: You will not get this panel if the selected database is a valid database, instead you will get an error message on the previous selection panel.

Select “2” (EXPAND) to bring up the CSS Utilities - EXPAND Shared Directory screen (see Figure 3-33).

Figure 3-33. CSS Utilities - EXPAND Shared Directory

```
------  COMPUWARE CSS UTILITIES - EXPAND Shared Directory ------ CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


Shared Directory: 'CX.LP.SHRDIR'
          Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)

   Model DSN ===>


     Space Units ===>                 (Blocks/Tracks/Cylinders)
        Quantity ===>

          CISIZE ===>



           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated, based on the fields entered above.

Select 3 (DELETE) to bring up the CSS Utilities - DELETE Shared Directory screen (see Figure 3-34).

Figure 3-34. CSS Utilities - Delete Shared Directory

```
------  COMPUWARE CSS UTILITIES - DELETE Shared Directory ------ CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)



Shared Directory: 'CX.LP.SHRDIR'
          Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)

                                                            DEFAULT:
   Attached Databases ===> DETACH          (Delete/Detach)    DETACH


        NOTE: Specifying DELETE will delete ALL databases attached
              to the shared directory



           Press ENTER to process  or  enter END command to terminate
```

This panel will generate a batch utility job to delete the shared directory, and will also either detach or delete all attached databases, according to the fields entered above.

Select “4” (VERIFY) to bring up the CSS Utilities - VERIFY Shared Directory screen (Figure 3-35).

Figure 3-35. CSS Utilities - VERIFY Shared Directory

```
------  COMPUWARE CSS UTILITIES - VERIFY Shared Directory ------ CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)



Shared Directory: 'CX.LP.SHRDIR'
          Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)









           Press ENTER to process  or  enter END command to terminate
```

This will generate a batch utility job to verify the shared directory and database integrity.

Working with Source Listing Database Files (File Manipulation)

Figure 3-36. Source Listing Source Listing Database - Advanced Functions - Main Panel

```
----------  Database  -  ADVANCED FUNCTIONS  --------------CSS V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Database: 'CX.LP.DATABASE'                                Volser: PRD925


          1  ATTACH       - Attach database to a shared directory
          2  DETACH       - Detach database from the shared directory
          3  CHANGE       - Change the Autodelete (AD) attribute
          4  DELETE       - Delete the database
          5  RESET        - Clear the shared directory information; only
                             to be used if the shared directory has been
                             damaged



            Press ENTER to process or enter END command to terminate
```

Select 1 (ATTACH) to bring up the CSS Utilities - ATTACH Database screen (Figure 3-37).

Figure 3-37. CSS Utilities - ATTACH Database

```
----------  COMPUWARE CSS UTILITIES - ATTACH Database ---------  CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


     Database: 'CX.LP.DATABASE'                                Volser: PRD971

          Preparation ===> EDITJCL         (Batch/Editjcl)


     Attach to
  Shared Directory ===>



           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated to attach the specified database to the specified shared directory.

Note: You will not get this panel if the selected database was already attached to a shared directory, instead you will get an error message on the main selection panel.

Select 2 (DETACH) to bring up the CSS Utilities - DETACH Database screen (Figure 3-38).

Figure 3-38. CSS Utilities - DETACH Database

```
----------  COMPUWARE CSS UTILITIES - DETACH Database ----------  CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


     Database: 'CX.LP.DATABASE'                                Volser: PRD925

  Attached to: 'CX.LP.SHRDIR'                                  Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)





           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated based on the fields entered to detach the above database from the specified shared directory.

Note: You will not get this panel if the selected database is already detached, instead you will get an error message on the main selection panel.

Select 3 (CHANGE) to bring up the CSS Utilities - CHANGE AUTODELETE screen (Figure 3-39).

Figure 3-39. CSS Utilities - CHANGE AUTODELETE

```
--------  COMPUWARE CSS UTILITIES - CHANGE AUTODELETE  ------- CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

     Database: 'CX.LP.DATABASE'                                Volser: PRD925

  Attached to: 'CX.LP.SHRDIR'                                  Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)

   Current Autodelete ===> YES

       New Autodelete ===>               (YES/NO/DUPS/STAGED)



           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated based on fields entered to change the autodelete setting of the specified database.

Select 4 (DELETE) to bring up the CSS Utilities - DELETE Database screen (Figure 3-40).

Figure 3-40. CSS Utilities - DELETE Database

```
----------  COMPUWARE CSS UTILITIES - DELETE Database ----------  CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


     Database: 'CX.LP.DATABASE'                                Volser: PRD925

  Attached to: 'CX.LP.SHRDIR'                                  Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)





           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated based on the fields entered that will first detach the specified database from its shared directory, then the database will be deleted.

Select 5 (RESET) to bring up the CSS Utilities - RESET Database screen (Figure 3-41).

Figure 3-41. CSS Utilities - RESET Database

```
----------  COMPUWARE CSS UTILITIES - RESET Database ----------  CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


     Database: 'CX.LP.DATABASE'                                Volser: PRD973

  Attached to: 'CX.LP.SHRDIR'                                  Volser:

          Preparation ===> EDITJCL         (Batch/Editjcl)





           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated to “reset” the database, and the shared directory name will be removed. This should only be done if the shared directory was destroyed.

Note: This panel will not appear if the database is attached to an existing shared directory, you will get an error message on the previous selection panel instead.

AAREPORT - Working with Abend-AID Report Files

Abend-AID report files are DDIO, shared directory, and database files that contain one or more Abend-AID reports. CSS Utilities enables you to perform a number of Abend-AID report file functions, including:

- Creating and formatting Abend-AID report files
- Reformatting existing Abend-AID report files
- Displaying Abend-AID report files
- Copying, moving, importing, and exporting Abend-AID report file members.
- File manipulation (for example, file deletion, verify shared directory, attached database list (for shared directory), change database auto-delete characteristics, etc.).

To work with Abend-AID report files, select AAREPORT from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2). The CSS Utilities - Abend-AID Report File Facility screen (Figure 3-42) is displayed.

Figure 3-42. CSS Utilities - Abend-AID Report File Facility Screen

```
--------------  CSS UTILITIES - Abend-AID Report File Facility  ----- V8.0.0
COMMAND ===>
Commands: SEtup   DDIOCALC (DDIO calculator)
                  DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

Processing Options for S/Member List:
    Confirm Delete ===> YES (Yes or No)
  Show print setup ===> YES (Yes or No)
          Language ===> ENGLISH           (ENGLISH/JAPANESE/USAMIX/USAUC)

SEL Options:  C (Create)  I (Info)  M (Move)  S (Member List)  A (Adv Function)

    SEL     --------------Abend-AID Report File(s)------------------------

     _      ______________________________________________________________
     _  (1)  'CX.TECH.TEST.DB80'
     _  (2)  'CX.REPTFILE'
     _  (3)  'CX.TECH.TEST.SDIR'
     _  (4)  'CX.ABENDAID.DATABASE'
     _  (5)  'CX.ABENDAID.SHRDIR'
     _  (6)  'CX.CREATE.NEW.FILE'
     _  (7)  'CX.LP.SRCEDIR'
     _  (8)
```

Setting Abend-AID Report File Options

The CSS Utilities - Abend-AID Report File Facility screen (Figure 3-42) enables you to set a number of Abend-AID report file options, described in Table 3-11. These options apply to the member list that is displayed when the **S (Member List)** option is selected.

For a listing of advanced parameters, see “Setting Advanced Parameters” on page 3-29.

Table 3-11. Abend-AID Report File Options

To:	Do This:
Edit the CSS Utilities General Settings	Enter **SE** at the Command prompt
Invoke the DDIOCALC function.	Enter **DDIOCALC** at the Command prompt

Table 3-11. Abend-AID Report File Options

To:	Do This:
Specify whether a confirmation screen will be displayed before deleting members	Enter one of the following in the Confirm Delete field: **Y:** Displays a confirmation screen before deleting a member from the Abend-AID report file. **N:** Deletes a member without displaying a confirmation screen.
Specify whether hardcopy options will be displayed before a hardcopy of an Abend-AID report file member is generated	Enter one of the following in the Show Print Setup field: **Y:** Displays hardcopy options before creating a hardcopy. **N:** Generates a hardcopy without displaying hardcopy options.
Specify the language in which the member list will be displayed when **S (member list)** is selected	Enter one of the following languages in the Language field: **English:** (Default) Displays the member list in mixed-case English. **Japanese:** Displays the member list in Japanese. Requires a terminal that supports DBCS. **USAMIX:** Displays the member list in mixed-case English. **USAUC:** Displays the member list in upper-case English.

Creating and Formatting Abend-AID Report Files

On the CSS Utilities - Abend-AID Report File Facility screen (Figure 3-42 on page 3-37), the Abend-AID Report Files section lists the eight Abend-AID report files that have been most recently used. To create and format Abend-AID report files, complete the following steps.

1. Perform the appropriate step:
   - To reformat an existing Abend-AID report file, scan the Abend-AID Report Files list. If it does not contain the file you want to reformat, enter the file name on the blank line at the top of the list. Then enter **C** to the left of the file name.
   - To create a new Abend-AID report file (and add the file to the list), enter the file name at the entry line under **Abend-AID Report Files**, then enter a **C** to the left of the file name.
2. Press **Enter**.

   If the file name you entered does NOT exist yet, you will first see the following file-type selection panel. You must select which type of file you want to create. Based on the type selected, one of the following “create” screens will then appear:

Figure 3-43. CSS Utilities - Select Desired File Type

```
--------CSS UTILITIES - Select Desired File Type -------CSS V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  File Name entered: 'CX.NEW.FILE'


          1  Shared Directory    - Create a shared directory
                                   NOTE: At least one database file must be
                                   attached to this shared directory for it
                                   to be functional.


          2  Database File       - Create a DDIO/Database that will be
                                   attached to an EXISTING shared directory

          3  DDIO File           - Create a non-database DDIO file


                      Enter END command to terminate
```

The CSS Utilities - Create/Format Abend-AID Report screen (Figure 3-44) is displayed for existing standard DDIO files, CSS Utilities - Create/Format Shared Directory (Figure 3-45) is displayed for exiting shared directory files, and CSS Utilities - Create/Format Report Database (Figure 3-46 on page 3-40) is displayed for existing Report Database files attached to a shared directory.

Figure 3-44. CSS Utilities - Create/Format Abend-AID File Screen (Standard DDIO Files)

```
--- COMPUWARE CSS UTILITIES - CREATE/FORMAT Abend-Aid Report File -- V7.9.0
COMMAND ===>

Commands: SEtup    A (Advanced Parms)    D (Delete file)
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

       Dataset Name ===> 'CX.REPTFILE'

        Preparation ===> EDITJCL    (Batch/Editjcl)
    DS Organization ===> SEQUENTIAL (Vsam/Sequential)
         Reallocate ===> NO         (Yes/No)

          Model DSN ===>
      Storage Class ===>            (optional)
   Management Class ===>            (optional)
         Data Class ===>            (optional)
             Volume ===>            (optional)
               Unit ===>            (optional)

        Space Units ===> TRACKS     (Blocks/Tracks/Cylinders)
   Primary Quantity ===> 200
  Number of Members ===> 100        (1 to 32767)

           Press ENTER to process  or  enter END command to terminate
```

Figure 3-45. CSS Utilities - Create/Format Abend-AID Shared Directory Files Screen

```
--- COMPUWARE CSS UTILITIES - CREATE/FORMAT Shared Directory ----- CSS V7.9.0
COMMAND ===>
Commands: SEtup    A (Advanced Parms)
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

       Dataset Name ===> 'CX.NEWFILE'

        Preparation ===> EDITJCL    (Batch/Editjcl)
         Reallocate ===> YES        (Yes/No)

          Model DSN ===> 'CX.NEW.AA.SD'
      Storage Class ===>            (optional)
   Management Class ===>            (optional)
         Data Class ===>            (optional)
             Volume ===>            (optional)

        Space Units ===> TRACKS     (Blocks/Tracks/Cylinders)
   Primary Quantity ===> 100
            OR
  Number of Entries ===> 100        (1 to 32767)      (optional)
  (This field overrides above space parms & calculates space needed)

           Press ENTER to process  or  enter END command to terminate
```

Figure 3-46. CSS Utilities - Create/Format Abend-AID Report Database File

```
--- COMPUWARE CSS UTILITIES - Create/Format Report Database - CSS V7.9.0
COMMAND ===>

Commands: SEtup    A (Advanced Parms)
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

       Dataset Name ===> 'CX.TECH.TEST.DATABASE'

        Preparation ===> EDITJCL    (Batch/Editjcl)
    DS Organization ===> V          (Vsam/Sequential)
         Reallocate ===> YES        (Yes/No)

          Model DSN ===>
      Storage Class ===>            (optional)
   Management Class ===>            (optional)
         Data Class ===>            (optional)
             Volume ===>            (optional)
               Unit ===>            (optional)

        Space Units ===> CYLINDERS  (Blocks/Tracks/Cylinders)
   Primary Quantity ===> 10
  Number of Members ===> 100        (1 to 32767)
   Shared directory ===> 'CX.ABENDAID.SHRDIR'

          Press ENTER to process  or  enter END command to terminate
```

3. Complete the fields on this screen using the following table as a guide.

Table 3-12. Create/Format Abend-AID Report File Fields

To:	**Do This:**
Edit the CSS Utilities General Settings	Enter **SE** at the Command prompt.
Display advanced parameters	Enter **A** at the Command prompt. For a complete description of the advanced parameters, see Table 3-13 on page 3-44.
Delete the Abend-AID report file specified in the Dataset Name field	Enter **D** at the Command prompt. *(Only valid for standard DDIO create)*

Table 3-12. Create/Format Abend-AID Report File Fields

To:	Do This:
Use batch processing to submit your job	Enter **BATCH** in the Preparation field.
Generate JCL for the Create/Format job, then edit the JCL	Enter **EDITJCL** in the Preparation field.
Specify the file type	Enter **VSAM** or **SEQUENTIAL** in the DS Organization field. **Note:** If a Model DSN is specified (see below), this parameter is optional. If a Model DSN is not specified, you must specify a file type.
Delete and reallocate the specified report file	Enter **YES** in the Reallocate field.
Specify a "template" (another Abend-AID report file) for allocating and formatting Abend-AID report files	**(Optional)** Enter the dataset name in the Model DSN field. **Note:** All explicitly set parameters on this screen, and on the Advanced Parameters screen, override settings in the template file.
Specify a Storage Management Subsystem (SMS) storage class	**(Optional)** Enter an eight-character storage class in the Storage Class field.
Specify an SMS management class	**(Optional)** Enter an eight-character management class in the Management Class field.
Specify an SMS data class	**(Optional)** Enter an eight-character data class in the Data Class field.
Specify the volume serial number of the Abend-AID report file	**(Optional)** Enter the volser in the Volume field.
Specify a device or device type	**(Optional)** Enter the device or device type in the Unit field.
Specify a unit of measure for the file size	Enter **BLOCKS**, **TRACKS**, or **CYLINDERS** in the Space Units field. **Note:** If DS Organization = VSAM and a Model DSN is specified, this parameter is optional. If not, this parameter is required.
Specify an amount of primary space for the Abend-AID report file	Enter a value in the Primary Quantity field. **Note:** If DS Organization = VSAM and a Model DSN is specified, this parameter is optional. If not, this parameter is required.
(DDIO or Database only) Specify the maximum number of members that the Abend-AID report file can contain	Enter a value in the Number of Members field. **Note:** If a Model DSN is specified, this parameter is optional. If not, this parameter is required.
(Shared Directory only) Specify the number of entries, Total number of members, to be stored In the shared directory.	**(Optional)** Enter a value in the Number of Entries field. If a value is entered here, it will override the space Parameters entered.
(Report Database Only) Specify the shared directory	Enter the name of the Shared directory you want the Database attached to.

4. After you have finished editing the fields, perform the appropriate action:
   - To submit the Create/Format job, ensure that the Command prompt is blank and press **Enter**.
   - To cancel the changes and return to the previous screen, enter **END** at the Command prompt.

Setting Advanced Create/Format Parameters

To edit Abend-AID report file Create/Format advanced parameters, complete the following steps:

1. Enter **A** at the command prompt on the CSS Utilities - Create/Format Abend-AID Report File options screen (Figure 3-44 on page 3-39).
2. Make sure the DS Organization field is set to the desired file type.
3. Press **Enter**.

Note: Figure 3-47 displays the Abend-AID Report File Create/Format - Advanced Parameters screen for Sequential Abend-AID report files. Figure 3-48 displays the Abend-AID Report File Create/Format - Advanced Parameters screen for VSAM Abend-AID report files.

Figure 3-47. Abend-AID Report File Create/Format - Advanced Parms (Sequential)

```
----- Abend-Aid REPORT FILE CREATE/FORMAT - ADVANCED PARAMETERS ---- V7.9.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Prompt this panel upon process ===> N  (Y/N)

  Enter/Modify parameters for Sequential Abend-Aid Report file:

                                                   DEFAULTS:
   Blocksize ===>         (2048 to TRKSIZE)           4096




  Groupcount ===>         (1 to 2048)                   4
  Autodelete ===> YES     (Dups/Yes/No)               YES
     Extents ===>         (1 to 460)                   64


  Note: Once any of the above parameters are modified, the new values will
        remain in effect for future formatting until they are modified again.

         Press ENTER to process  or  enter END command to terminate
```

Figure 3-48. Abend-AID Report File Create/Format - Advanced Parms (VSAM)

```
----- Abend-AID Report File CREATE/FORMAT - ADVANCED PARAMETERS -----V7.9.0
 COMMAND ===>
 Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

   Prompt this panel upon process ===> N  (Y/N)

   Enter/Modify parameters for VSAM Abend-AID Report file:

                                                         DEFAULTS:
          CISIZE ===>                                        4096
       Secondary ===>         (Yes or No)                     NO
    Sec.Quantity ===>         (Uses primary space units)
    VSAM Catalog ===>

      Groupcount ===>         (1 to 2048)                      4
      Autodelete ===> DUPS    (Dups/Yes/No)                  YES
         Extents ===>         (1 to 460)                      64


  Note: Once any of the above parameters are modified, the new values will
        remain in effect for future formatting until they are modified again.

         Press ENTER to process  or  enter END command to terminate
```

Figure 3-49. Abend-AID Report File Shared Directory Create - Advanced Parms

```
----- Shared Directory  CREATE/FORMAT - ADVANCED PARAMETERS -----CSS V7.9.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Prompt this panel upon process ===> N  (Y/N)

  Enter/Modify parameters for VSAM Shared Directory:

                                                        DEFAULTS:
          CISIZE ===> 8000                                 4096
       Secondary ===> YES     (Yes or No)                    NO

    VSAM Catalog ===>





 Note: Once any of the above parameters are modified, the new values will
       remain in effect for future formatting until they are modified again.

         Press ENTER to process or enter  END command to terminate
```

Figure 3-50. Abend-AID Report Database Create/Format - Advanced Parms (VSAM)

```
---- Report Database  Create/Format - Advanced Parameters --- CSS V7.9.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Prompt this panel upon process ===> N  (Y/N)

  Enter/Modify parameters for VSAM Report Database:


                                                        DEFAULTS:
          CISIZE ===>                                      4096
       Secondary ===> YES     (Yes or No)                    NO

    VSAM Catalog ===>

      Groupcount ===> 20      (1 to 2048)                     4
      Autodelete ===> YES     (Dups/Yes/No)                 YES
         Extents ===>         (1 to 460)                     64


 Note: Once any of the above parameters are modified, the new values will
       remain in effect for future formatting until they are modified again.
```

Figure 3-51. Abend-AID Report Database Create/Format - Advanced Parms (Sequential)

```
------ Report Database  Create/Format - Advanced Parameters ---- CSS V7.9.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Prompt this panel upon process ===> N  (Y/N)

  Enter/Modify parameters for Sequential Report Database:

                                                      DEFAULTS:
   Blocksize ===>         (2048 to TRKSIZE)             4096



  Groupcount ===> 20      (1 to 2048)                     4
  Autodelete ===> YES     (Dups/Yes/No)                 YES
     Extents ===>         (1 to 460)                     64



  Note: Once any of the above parameters are modified, the new values will
        remain in effect for future formatting until they are modified again.

         Press ENTER to process  or  enter END command to terminate
```

4. Complete the fields on this screen, as necessary, using the following table as a guide.

Table 3-13. Create/Format Abend-AID Report File Advanced Parameters

To:	Do This:
Specify whether advanced parameters will be displayed each time a Create/Format Abend-AID Report File command is processed	Enter one of the following in the Prompt This Panel Upon Process field: **Y:** Displays advanced parameters before processing Create/Format commands. **N:** Processes Create/Format commands without displaying advanced parameters.
(Sequential files only) Specify the Abend-AID report file blocksize	Enter an integer between 2048 and 32760 in the Blocksize field. • If you do not complete this field, the default blocksize of 4096 will be used. • The blocksize is always ignored during the file allocation phase.
(VSAM files only) Specify the control interval size	Enter an integer in the CISIZE field. If you do not complete this field, the default value of 4096 will be used.
(VSAM files only) Specify whether VSAM Abend-AID report files will be allocated with secondary extents	Enter one of the following in the Secondary field: **Yes:** VSAM Abend-AID report files will be allocated with secondary extents. **No:** (Default) VSAM Abend-AID report files will not be allocated with secondary extents.
(VSAM files only) Specify the amount of secondary space for the specified Abend-AID report file (used when the primary space has been consumed)	Enter a number of blocks, tracks, or cylinders in the Sec. Quantity field. **Note:** The unit of measure (blocks, tracks, or cylinders) is determined by the Space Units setting on the Create/Format Abend-AID Report File screen.
(VSAM files only) Specify the VSAM catalog in which the Abend-AID report file will be created	**(Optional)** Enter a catalog name in the VSAM Catalog field.

Table 3-13. Create/Format Abend-AID Report File Advanced Parameters

To:	Do This:
Specify the number of blocks allocated to an Abend-AID report file when additional space is needed	Enter a number between 1 and 2048 in the Groupcount field. **Note:** Compuware recommends setting the Groupcount to **8**. The default setting is **4**.
Specify whether the oldest member of an Abend-AID report file will be deleted when the file becomes full	Enter one of the following settings in the Autodelete field: **Dups:** Unlocks automatically locked members with the same name, then locks the most recent member. **Yes:** (Default) Automatically deletes one or more of the oldest unlocked members to allow new members to be written to the file. **No:** You must manually delete members when the DDIO file becomes full. For a complete description of these settings, see "Command Parameters for DDIO Files" on page 4-6.
Specify the maximum number of extents allocated to each Abend-AID report file member	Enter a number between 1 and 460 in the Extents field. If you do not complete this field, the default value of 64 will be used.

5. After you have finished editing the fields, perform the appropriate action:
   - To apply the advanced parameter settings and return to the previous screen, ensure that the Command prompt is blank and press Enter.
   - To cancel the changes and return to the previous screen, enter **END** at the Command prompt.

Deleting Abend-AID Report Files

To delete an Abend-AID report file, complete the following steps.

1. Ensure that the CSS Utilities - Abend-AID Report File Facility screen (Figure 3-42 on page 3-37) is displayed.

   You can display this screen by selecting option 3 (AAREPORT) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).

2. Scan the Abend-AID Report Files list on this screen. If it does not contain the file that you want to delete, enter the file name on the blank line at the top of the list.

For standard DDIO files only - Steps 3 - 7:

3. Enter **C** to the left of the file name.
4. Press **Enter**.

   The CSS Utilities - Create/Format Abend-AID Report Files screen (Figure 3-44 on page 3-39) is displayed.

5. Ensure that the correct Abend-AID report file is specified in the Dataset Name field.
6. Enter **D** at the command prompt.
7. Press **Enter**.

For all types of Abend-AID Report files - Steps 8 - 12:

8. Or, optionally, (instead of steps 3 through 7) enter **A** to the left of the file name
9. Press **Enter**.
10. The CSS Utilities Advanced Functions panel will be displayed.
11. Select the "Delete" option from the menu, press **Enter**.

12. Follow instructions on the particular “delete” function panel that is displayed (different by file type). See “Setting Advanced Parameters” on page 3-29.

Displaying File Information

To display information about a particular Abend-AID report file, complete the following steps:

1. Ensure that the CSS Utilities - Abend-AID Report File Facility screen (Figure 3-42 on page 3-37) is displayed.

   You can display this screen by selecting option 3 (AAREPORT) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).

2. Scan the DDIO Files list on this screen. If it does not contain the file that you want to view information about, enter the file name on the blank line at the top of the list.

3. Enter **I** to the left of the file name.

4. Press **Enter**.

   A report (Figure 3-52) is displayed, containing current parameter settings and file statistics. To scroll through the report, use the **PF7** and **PF8** keys.

Figure 3-52. Abend-AID Report File Information Report

```
COMPUWARE/VF ----- 'CX.REPTFILE' --------------------------- Row 1 to 21 of 41
COMMAND INPUT ===>                                            SCROLL ===> PAGE
                      DIRECTORY INFORMATION

 DSN ---------------------------- CX.REPTFILE
 VOLSER ------------------------- PRD933
 AM(ACCESS METHOD) --------------- BSAM
 BLKSIZE ------------------------- 8185
 TYPE ---------------------------- ABEND-AID
 REPTCOUNT ----------------------- 10
 GROUPCOUNT ---------------------- 4
 AUTODELETE ---------------------- DUPS
 INDEX --------------------------- EXT
 EXTENTS ------------------------- 256
 OTHER INFORMATION
   BLOCKS IN FILE ---------------- 360
   ALLOCATION GROUPS ------------- 89
   USED ALLOCATION GROUPS -------- 10(11%)
   BLOCKS/TRACK ------------------ 6
   FILE FORMATTED BY DDIO RELEASE- 7.4.0
   REQUIRED MINIMUM DDIO RELEASE - 5.6.0
   CURRENT REPORT NUMBER --------- 11
   DIRECTORY ENTRIES/BLOCK ------- 13
   DIRECTORY BLOCKS -------------- 1
```

Note: To see a complete report sample and a description of the fields on the report, refer to “Abend-AID Report File DIRX Report” on page 2-24.

Figure 3-53. Abend-AID Report Database Information Screen Format

```
COMPUWARE/VF ----- 'CX.AA.TEST.DB1' ------------------------ Row 1 to 21 of 44
COMMAND INPUT ===>                                            SCROLL ===> CSR
                        DIRECTORY INFORMATION
 DATABASE IS CONNECTED
 CATALOG DSN: CX.AA.TEST.SDIR
      VOLSER: PRD907

 DSN----------------------------- CX.AA.TEST.DB1
 VOLSER -------------------------- PRD948
 AM(ACCESS METHOD) --------------- BSAM
 BLKSIZE ------------------------- 4096
 TYPE ---------------------------- ABEND-AID
 REPTCOUNT ----------------------- 250
 GROUPCOUNT ---------------------- 4
 AUTODELETE ---------------------- NO
 INDEX --------------------------- EXT
 EXTENTS ------------------------- 64
 OTHER INFORMATION
   BLOCKS IN FILE ---------------- 7200
   ALLOCATION GROUPS ------------- 1796
   USED ALLOCATION GROUPS -------- 237(13%)
   BLOCKS/TRACK ------------------ 12
   FILE FORMATTED BY DDIO RELEASE- 7.9.0
   REQUIRED MINIMUM DDIO RELEASE - 7.0.0
```

Figure 3-54. Report Shared Directory Information Screen Format

```
COMPUWARE/VF ----- 'CX.AA.TEST.SDIR' ---------------------- Row 1 to 20 of 117
COMMAND INPUT ===>                                            SCROLL ===> CSR
                        DIRECTORY INFORMATION
 DIRX FOR SHARED DIRECTORY: CX.AA.TEST.SDIR
                    VOLSER: PRD907


 SHARED DIRECTORY INFORMATION:
 AM ------------------------------ VSAM
 BLKSIZE ------------------------- 18425
 OTHER INFORMATION
   DIRECTORY FORMATTED BY DDIO --- 7.8.0
   REQUIRES MINIMUM DDIO RELEASE - 7.0.0
   TOTAL BLOCKS IN SHARED DIR ---- 180
   USED BLOCKS ------------------- 67(37%)
   NUMBER OF EXTENTS ------------- 1


   DIRECTORY CLASS: DATABASE
   CLASS INSTANCE NAME ----------- DATABASE
   CLASS ANCHOR RBN -------------- 12
   NUMBER OF ENTRIES ------------- 10
   CURRENT PUT NUMBER ------------ 37
```

Working with Abend-AID Report File Members

CSS Utilities enables you to work with Abend-AID report file members in the following ways:

- Display members
- Copy members
- Move members
- Export members
- Import members

These operations are described in the following sections.

Displaying Members

To display the members in a particular Abend-AID report file, complete the following steps:

1. Ensure that the CSS Utilities - Abend-AID Report File Facility screen (Figure 3-42 on page 3-37) is displayed.

   You can display this screen by selecting option 3 (AAREPORT) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).

2. Scan the Abend-AID Report Files list on this screen. If it does not contain the desired file, enter the file name on the blank line at the top of the list.

3. Enter **S** to the left of the file name.

4. Press **Enter**.

   A member list (Figure 3-55) is displayed. To scroll through the list, use the **PF7** and **PF8** keys. To display a list of the line commands available on this screen, press **PF1**.

Figure 3-55. Abend-AID Report File Member List

```
COMPUWARE/VF ----- 'CX.REPTFILE' --------------------------- Row 1 to 10 of 10
COMMAND INPUT ===>                                            SCROLL ===> PAGE
                        Abend-AID DATASET DIRECTORY

      S - View C - Contact Info D - Delete P - Print L - Lock U - Unlock

   JOBNAME   REPT NUMBER  JESID CODE        DATE       TIME   DESC      SIZE(K)
  JOBABCR  (L)        10 J29906 SOCB     01 APR 2002 14.18.34 SMITH          32
  JOBABCR  *I          9 J21264 SOC1     31 MAR 2002  8.47.03 SMITH          32
  JOBABCB  (L)         8 J19762 SOCB     28 JAN 2002 16.27.11 SMITH,         32
  JOBABCB              7 J03229 SOCB     26 JAN 2002 13.16.24 SMITH,         32
  JOBABCB              6 J01638 SOCB     26 JAN 2002  9.20.35 SMITH,         32
  JOBABCB              5 J24405 SOCB     23 JAN 2002  9.05.15 SMITH,         32
  JOBABCB              4 J18937 SOCB     22 JAN 2002 11.31.48 SMITH,         32
  JOBABCB              3 J21640 SOCB     17 DEC 2001 15.03.29 SMITH,         32
  JOBABCB              2 J21549 SOCB     17 DEC 2001 14.57.06 SMITH,         32
  JOBABCB              1 J18260 SOCB     17 DEC 2001 10.27.03 SMITH,         32
****************************** Bottom of data ******************************
```

Note: To see more information about this portion of the report, refer to “Column Headings” on page 2-28.

Copying Members

To copy members from one Abend-AID report file to another, complete the following steps:

1. Ensure that the CSS Utilities - Abend-AID Report File Facility screen (Figure 3-42 on page 3-37) is displayed.

   You can display this screen by selecting option 3 (AAREPORT) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).

2. Scan the DDIO Files list on this screen. If it does not contain the file from which you want to copy members, enter the file name on the blank line at the top of the list.

3. Enter **M** to the left of the file name.

4. Press **Enter**.

   The CSS Utilities - Abend-AID Report File Move Facility screen (Figure 3-56) is displayed.

Figure 3-56. CSS Utilities - Abend-AID Report File Move Facility Screen

```
- COMPUWARE CSS UTILITIES - Abend-Aid Report File Move Facility - V8.0.0
COMMAND ===>

Commands: blank (Process)   SEtup   L (List of members)   A (Advanced parms)
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


   Preparation ===> EDITJCL         (Batch/Editjcl)

        Option ===> COPY            (Copy/Move/Export/Import)

     From File ===> 'CX.REPTFILE'
       To File ===> 'CX.NEW.REPTFILE'


   Member Name ===>                 (Optional, specify prefix to be used for
                                     select - can be a wildcard or entire name)

 Member Number ===>                 (Optional, specify a member number to be
                                     used - will override "Member Name" field)


           Press ENTER to process  or  enter END command to terminate
```

5. Complete the fields on this screen using the following table as a guide.

Note: This table describes the COPY, MOVE, EXPORT, and IMPORT commands. You will be using the COPY command, so specify COPY in the Option field.

Table 3-14. Abend-AID Report File Move Facility Fields

To:	Do This:
Use batch processing to submit your job	Enter **BATCH** in the Preparation field.
Edit the JCL generated for the COPY, MOVE, EXPORT, or IMPORT before submitting the JCL for processing	Enter **EDITJCL** in the Preparation field.
Specify the command type (COPY, MOVE, EXPORT, or IMPORT)	Enter the command at the Option prompt.
Specify the Abend-AID report file **from which** members will be copied, moved, exported, or the sequential file from which members will be imported	Enter a dataset name in the From File field.
Specify the Abend-AID report file **to which** members will be copied, moved, or imported	Enter a dataset name in the To File field. If this dataset does not exist, you will be prompted to enter the **C** command to display the CSS Utilities - Create/Format Abend-AID Report File screen (Figure 3-42 on page 3-37 if From File is a standard DDIO file, or Figure 3-57 on page 3-51 if From File is a shared directory or database).
Specify the sequential file **to which** members will be exported	Enter a dataset name in the To File field. If this dataset does not exist, it will be created as part of the generated EXPORT job in the JCL.
Perform commands (COPY, MOVE, IMPORT, EXPORT, and LIST) on members matching a particular prefix	Optional: Enter the prefix in the Member Name field. Prefix can be a wild card (followed by an asterisk), or the entire name. **Note:** If a prefix is not specified, commands will be performed on all members in the specified file.
Specify a particular source member for processing by member number	Enter the member number in the Member Number field.

Table 3-14. Abend-AID Report File Move Facility Fields

To:	Do This:
View and select members in the **From File** dataset	Enter **L** at the Command prompt. **Note:** See "Selecting Specific Members for Processing" on page 3-52 for step-by-step instructions.
Edit advanced parameters	Enter **A** at the Command prompt. For a complete description of the advanced parameters, see "Setting Advanced Parameters" on page 3-29.
Edit the CSS Utilities General Settings	Enter **SETUP** at the Command prompt.

6. After you have finished editing the fields, perform the appropriate action:
   - To submit the COPY, MOVE, IMPORT, or EXPORT, ensure that the Command prompt is blank and press **Enter**.
   - To cancel the changes and return to the previous screen, enter **END** at the Command prompt.

Moving Members

To move members from one Abend-AID report file to another, complete the following steps:

1. Ensure that the CSS Utilities - Abend-AID Report File Facility screen (Figure 3-42 on page 3-37) is displayed.

   You can display this screen by selecting option 3 (AAREPORT) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).
2. Scan the Abend-AID Report Files list on this screen. If it does not contain the file from which you want to move members, enter the file name on the blank line at the top of the list.
3. Enter **M** to the left of the file name.
4. Press **Enter**.

   The CSS Utilities - Abend-AID Report File Move Facility screen (Figure 3-56 on page 3-49) is displayed.
5. Complete the fields on this screen using Table 3-14 on page 3-49 as a guide.
   - Specify MOVE in the Option field.
   - Enter a dataset name in the To File field.

Exporting Members

To export members from an Abend-AID report file to a sequential dataset, complete the following steps:

1. Ensure that the CSS Utilities - Abend-AID Report File Facility screen (Figure 3-42 on page 3-37) is displayed.

   You can display this screen by selecting option 3 (AAREPORT) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).
2. Scan the Abend-AID Report Files list on this screen. If it does not contain the file from which you want to export members, enter the file name on the blank line at the top of the list.
3. Enter **M** to the left of the file name.
4. Press **Enter**.

The CSS Utilities - Abend-AID Report File Move Facility screen (Figure 3-56 on page 3-49) is displayed.

5. Complete the fields on this screen using Table 3-14 on page 3-49 as a guide.
   - Specify EXPORT in the Option field.
   - Enter the name of a sequential dataset in the To File field. If this dataset does not exist, it will be created as part of the generated EXPORT job in the JCL.

Importing Members

To import members from a sequential dataset into an Abend-AID report file, complete the following steps:

1. Ensure that the CSS Utilities - Abend-AID Report File Facility screen (Figure 3-42 on page 3-37) is displayed.

   You can display this screen by selecting option 3 (AAREPORT) from the CSS Utilities - Primary Menu (Figure 3-1 on page 3-2).
2. Enter the name of the file to import members **from** on the blank line at the top of the list.
3. Enter **M** to the left of the file name.
4. Press **Enter**.

   The CSS Utilities - Abend-AID Report File Move Facility screen (Figure 3-56 on page 3-49) is displayed.
5. Complete the fields on this screen as shown in Table 3-14 on page 3-49.
   - Specify IMPORT in the Option field.
   - Enter the name of an Abend-AID report file DDIO dataset in the To File field.

Specifying "C" to Create the "TO" File for Shared Directory or Database Files

If you are prompted to enter the "C" command to create the file name specified on the "TO file" field, and if the "FROM file" is either a shared directory or a database file, this file type selection panel will be displayed. You may create either a database that will be attached to an existing shared directory, or you may create a new shared directory and then also must create a database attached to it. For Abend-AID, see Figure 3-45 and Figure 3-46 on page 3-40.

Figure 3-57. Select Desired File Type

```
--------CSS UTILITIES - Select Desired File Type -------CSS V7.9.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)


  File Name entered: 'CX.NEW.LP.SD'


          1  Shared Directory    - Create a shared directory
                                   NOTE: You must also next create at least
                                         one database file that will be
                                         attached to this shared directory.


          2  Database File       - Create a DDIO/Database that will be
                                   attached to an EXISTING shared directory.



                      Enter END command to terminate
```

Selecting Specific Members for Processing

To copy, move, or import specific members within a dataset, complete the following steps:

1. Enter **L** at the command prompt on the CSS Utilities - Abend-AID Report File Move Facility screen (Figure 3-56 on page 3-49).

   A list containing all members in the **From File** dataset (Figure 3-58) is displayed.

Figure 3-58. Selecting Members for COPY, MOVE, or IMPORT

```
COMPUWARE/VF ----- 'CX.REPTFILE' ---------------- V8.0.0 --- Row 1 to 10 of 10
COMMAND INPUT ===>                                          SCROLL ===> PAGE

       Line Commands:  S (Select for processing)
                      Abend-Aid REPORT DATASET DIRECTORY
   JOBNAME   REPT NUMBER  JESID CODE        DATE       TIME   DESC       IZE(K)
  JOBABCR  (L)        10 J29906 SOCB     01 APR 1998 14.18.34 SMITH          32
  JOBABCR  *I          9 J21264 SOC1     31 MAR 1998  8.47.03 SMITH          32
  JOBABCB  (L)         8 J19762 SOCB     28 JAN 1998 16.27.11 SMITH,         32
  JOBABCB              7 J03229 SOCB     26 JAN 1998 13.16.24 SMITH,         32
  JOBABCB              6 J01638 SOCB     26 JAN 1998  9.20.35 SMITH,         32
  JOBABCB              5 J24405 SOCB     23 JAN 1998  9.05.15 SMITH,         32
  JOBABCB              4 J18937 SOCB     22 JAN 1998 11.31.48 SMITH,         32
  JOBABCB              3 J21640 SOCB     17 DEC 1997 15.03.29 SMITH,         32
  JOBABCB              2 J21549 SOCB     17 DEC 1997 14.57.06 SMITH,         32
  JOBABCB              1 J18260 SOCB     17 DEC 1997 10.27.03 SMITH,         32
****************************** Bottom of data ******************************
```

2. Select members by entering an **S** to the left of the member names.

   Selected members are marked with an asterisk and the word **SELECTED**.

3. Press **F3** to return to the CSS Utilities - Abend-AID Report File Move Facility screen (Figure 3-56 on page 3-49).

4. Ensure that the command prompt is blank, then press **Enter**.

Setting Advanced Parameters

To set Abend-AID report file advanced parameters, complete the following steps:

1. Ensure that the CSS Utilities - Abend-AID Report File Move Facility screen (Figure 3-56 on page 3-49) is displayed.

2. Enter **A** at the command prompt.

   The Abend-AID Report File Move Function- Advanced Parameters screen (Figure 3-59) is displayed.

Figure 3-59. Abend-AID Report File Move Function - Advanced Parameters Screen

```
----- Abend-Aid Report File Move Function - ADVANCED PARAMETERS ---- V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Prompt this panel upon process ===> N  (Y/N)

  Enter/Modify parameters for COPY/MOVE/EXPORT/IMPORT:


                                                          DEFAULTS:
          Age old ===>             nnn
      Age Units ===>             (DAYS/MONTHS/YEARS)    DAYS


       CONTINUE ===>             (YES/NO)               YES
         DELETE ===>             (NO/YES)               NO
           EXEC ===>             (SUMMARY/CHECK/RUN)    SUMMARY
          LABEL ===>             (1 to nn)              1

  Note: Once any of the above parameters are modified, the new values will
        remain in effect for future formatting until they are modified again.

         Press ENTER to process  or  enter END command to terminate
```

3. Edit the parameters as needed, using Table 3-15 as a guide.

Table 3-15. Abend-AID Report File Advanced Parameters

To:	Do This:
Specify whether advanced parameters will be displayed each time a COPY, MOVE, EXPORT, IMPORT, or LIST command is processed	Enter one of the following at the Prompt This Panel Upon Process field: **Y:** Displays advanced parameters before processing COPY, MOVE, EXPORT, IMPORT, and LIST commands. **N:** Processes these commands without displaying advanced parameters.
Restrict the COPY, MOVE, EXPORT, and IMPORT commands to members older than **nnn** months, days, or years	1. Specify a number in the Age Old field. 2. Specify a unit (**months**, **days**, or **years**) in the Age Units field.
Specify whether COPY, MOVE, EXPORT, and IMPORT commands will continue if a recoverable error is detected	Enter one of the following in the CONTINUE field: **Y:** (Default) Command continues if possible when an error is detected. **N:** Command stops when an error is detected.
Specify whether members will be deleted from the original file (the **From File**) after they are successfully exported	Enter one of the following in the DELETE field: **Y:** Deletes members after they are exported. **N:** (Default) Does not delete members after they are exported.
Control how the COPY, MOVE, EXPORT, and IMPORT commands are processed (Not present for Database and/or Shared Directory files)	Enter one of the following in the EXEC field: **SUMMARY:** (Default) Transfers data and produces a summary report. The summary includes a list of transferred members and a list of any members deleted from the From File during the transfer. **CHECK:** Produces a detailed report without moving any data. This report includes a list of transferred members, a list of any members deleted from the From File during the transfer, and a list of attributes from the To and From files. It also includes a list of any members unlocked or deleted during MOVE, COPY, or IMPORT commands. **RUN:** Transfers data and produces a detailed report. See CHECK (above) for a description of the report contents.

Table 3-15. Abend-AID Report File Advanced Parameters

To:	Do This:
Reference multiple export files using a single ddname — create export files on a single tape	Enter the file number of a sequential dataset in the LABEL field. This parameter will be used with the IMPORT and EXPORT commands. You can include the LABEL parameter in your JCL. However, a LABEL parameter specified on this panel overwrites any LABEL setting in your JCL.

4. After you have finished editing the parameters, perform the appropriate action:
   - To apply the advanced parameter settings and return to the previous screen, ensure that the Command prompt is blank and press **Enter**.
   - To cancel the changes and return to the previous screen, enter **END** at the command prompt.

Advanced Functions - Abend-AID

Working with Abend-AID Report Files (File Manipulation)

Figure 3-60. Standard DDIO Advanced Function Panel - Abend-AID Report File - Main Panel

```
------ AbendAID Report File - ADVANCED FUNCTIONS ----- CSS V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

 AbendAID DDIO File: 'CX.REPTFILE'
             Volser: PRD964



          1  DELETE       - Delete the AbendAID DDIO file




            Press ENTER to process or enter END command to terminate
```

This panel appears after entering selection “1”;

Figure 3-61. CSS Utilities - CONFIRM Delete

```
------------------  CSS UTILITIES CONFIRM DELETE   -------------------- V8.0.0
COMMAND ===>

Dataset Name: CX.REPTFILE


Press ENTER to confirm delete process or enter END to cancel this request.
```

Note: This performs an immediate dynamic file delete, just like the original “D” delete function on the create panel.

Working with Report Shared Directory Files (File Manipulation)

Figure 3-62. Abend-AID Report File Shared Directory - Advanced Functions - Main Panel

```
--------- Shared Directory  -  ADVANCED FUNCTIONS  --------- CSS V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

Shared Directory: 'CX.TECH.TEST.SDIR'
          Volser: PRD907

          1  LISTDB       - List all attached databases
          2  EXPAND       - Expand size of the shared directory
          3  DELETE       - Delete the shared directory
          4  VERIFY       - Verify the integrity of the shared directory




            Press ENTER to process or enter END command to terminate
```

Select 1 (LISTDB) to bring up the Shared Directory - Attached Database Report screen (Figure 3-63).

Figure 3-63. Shared Directory - Attached Database Report

```
COMPUWARE/VF --- Shared Directory - Attached Database Report - Row 1 to 3 of 3
COMMAND INPUT ===>                                          SCROLL ===> CSR
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

Shared Directory: 'CX.TECH.TEST.SHRDIR'
          Volser: PRD988
 I-info  C-change  T-detach  D-delete  R-scratch  S-list member  M-move member

SEL  --------------- Database List --------------    Volser
  _   CX.TECH.TEST.DB1
  _   CX.TECH.TEST.DB2
  _   CX.TECH.TEST.DB3
****************************** Bottom of data ********************************
```

- Select **I** to display the database “display file information” screen (Figure 3-53 on page 3-47).
- Select **C** to display the database “advanced functions - change” screen (Figure 3-71 on page 3-59).
- Select **T** to display the database “advanced functions - detach” screen (Figure 3-70 on page 3-59).
- Select **D** to display the database “advanced functions - delete” screen (Figure 3-72 on page 3-60).
- Select **R** to display the “CSS Utilities - SCRATCH Database” screen (Figure 3-64).
- Select **S** to display the database “member list” screen (Figure 3-55 on page 3-48).
- Select **M** to display the database “move member” screen (Figure 3-56 on page 3-49).

Note: Pressing **PF3** from any of these screens will return you to the Shared Directory - Attached Database Report screen (Figure 3-63).

Select **R** (SCRATCH) to bring up the CSS Utilities - SCRATCH Database screen (Figure 3-64).

Figure 3-64. CSS Utilities - SCRATCH Database

```
----------  COMPUWARE CSS UTILITIES - SCRATCH Database ----------  CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


     Database: 'CX.TECH.TEST.DB1'                             Volser: PRD988

  Attached to: 'CX.TECH.TEST.SHRDIR'                          Volser:

          Preparation ===> EDITJCL         (Batch/Editjcl)





           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated to do the “scratch” function — removing the specified database from this shared directory. This should only be done when the database has been destroyed.

Note: You will not get this panel if the selected database is a valid database, you will get an error message on the previous selection panel instead.

Select “2” (EXPAND) to bring up the CSS Utilities - EXPAND Shared Directory screen (Figure 3-65).

Figure 3-65. CSS Utilities - EXPAND Shared Directory

```
------  COMPUWARE CSS UTILITIES - EXPAND Shared Directory ------ CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


Shared Directory: 'CX.TECH.TEST.SHRDIR4'
          Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)

   Model DSN ===>


     Space Units ===>                 (Blocks/Tracks/Cylinders)
        Quantity ===>

          CISIZE ===>



           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated, based on the fields entered above.

Select “3” (DELETE) to bring up the CSS Utilities - DELETE Shared Directory screen (Figure 3-66).

Figure 3-66. CSS Utilities - DELETE Shared Directory

```
------  COMPUWARE CSS UTILITIES - DELETE Shared Directory ------ CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)



Shared Directory: 'CX.TECH.TEST.SHRDIR'
          Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)

                                                            DEFAULT:
   Attached Databases ===> DETACH          (Delete/Detach)    DETACH


        NOTE: Specifying DELETE will delete ALL databases attached
              to the shared directory



           Press ENTER to process  or  enter END command to terminate
```

This panel will generate a batch utility job to delete the shared directory, and will also either detach or delete all attached databases, according to the fields entered above.

Select “4” (VERIFY) to bring up the CSS Utilities - VERIFY Shared Directory screen (Figure 3-67).

Figure 3-67. CSS Utilities - VERIFY Shared Directory

```
------  COMPUWARE CSS UTILITIES - VERIFY Shared Directory ------ CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)



Shared Directory: 'CX.TECH.TEST.SHRDIR'
          Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)









           Press ENTER to process  or  enter END command to terminate
```

This will generate a batch utility job to verify the shared directory and database integrity.

Working with Abend-AID Report Database Files (File Manipulation)

Figure 3-68. Abend-AID Report File Report Database - Advanced Functions - Main Panel

```
----------  Database  -  ADVANCED FUNCTIONS  --------------CSS V8.0.0
COMMAND ===>
Commands: SEtup  DD (DDIO tutorial)  SD (SD/DB tutorial)  LP (LP tutorial)

  Database: 'CX.TECH.TEST.DB98'                            Volser: PRD954


          1  ATTACH       - Attach database to a shared directory
          2  DETACH       - Detach database from the shared directory
          3  CHANGE       - Change the Autodelete (AD) attribute
          4  DELETE       - Delete the database
          5  RESET        - Clear the shared directory information; only
                             to be used if the shared directory has been
                             damaged



            Press ENTER to process or enter END command to terminate
```

Select “1” (ATTACH) to bring up the CSS Utilities - ATTACH Database screen (Figure 3-69).

Figure 3-69. CSS Utilities - ATTACH Database

```
----------  COMPUWARE CSS UTILITIES - ATTACH Database ---------  CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)



     Database: 'CX.TECH.TEST.DATABASE'                       Volser: PRD971

          Preparation ===> EDITJCL         (Batch/Editjcl)


     Attach to
  Shared Directory ===>



           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated to attach the specified database to the specified shared directory.

Note: You will not get this panel if the selected database was already attached to a shared directory - you will get an error message on the main selection panel instead.

Select “2” (DETACH) to bring up the CSS Utilities - DETACH Database screen (Figure 3-70).

Figure 3-70. CSS Utilities - DETACH Database

```
----------  COMPUWARE CSS UTILITIES - DETACH Database ----------  CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


     Database: 'CX.TECH.TEST.DATABASE'                        Volser: PRD925

  Attached to: 'CX.TECH.TEST.SHRDIR'                          Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)





           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated based on the fields entered to detach the above database from the specified shared directory.

Note: You will not get this panel if the selected database is already detached, you will get an error message on the main selection panel instead.

Select “3” (CHANGE) to bring up the CSS Utilities - CHANGE AUTODELETE screen (Figure 3-71).

Figure 3-71. CSS Utilities - CHANGE AUTODELETE

```
--------  COMPUWARE CSS UTILITIES - CHANGE AUTODELETE  ------- CSS 8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

     Database: 'CX.TECH.TEST.DATABASE'                        Volser: PRD925

  Attached to: 'CX.TECH.TEST.SHRDIR'                          Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)

   Current Autodelete ===> YES

       New Autodelete ===>               (YES/NO/DUPS)



           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated based on fields entered to change the autodelete setting of the specified database.

Select “4” (DELETE) to bring up the CSS Utilities - DELETE Database screen (Figure 3-72).

Figure 3-72. CSS Utilities - DELETE Database

```
----------  COMPUWARE CSS UTILITIES - DELETE Database ----------  CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


     Database: 'CX.TECH.TEST.DATABASE'                        Volser: PRD925

  Attached to: 'CX.TECH.TEST.SHRDIR'                          Volser: PRD988

          Preparation ===> EDITJCL         (Batch/Editjcl)





           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated based on the fields entered that will first detach the specified database from its shared directory, then the database will be deleted.

Select “5” (RESET) to bring up the CSS Utilities - RESET Database.

Figure 3-73. CSS Utilities - RESET Database

```
----------  COMPUWARE CSS UTILITIES - RESET Database ----------  CSS V8.0.0
COMMAND ===>
Commands: blank (Process)      SEtup
          DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)


     Database: 'CX.TECH.TEST.DATABASE'                        Volser: PRD973

  Attached to: 'CX.TECH.TEST.SHRDIR'                          Volser:

          Preparation ===> EDITJCL         (Batch/Editjcl)





           Press ENTER to process  or  enter END command to terminate
```

A batch utility job will be generated to “reset” the database — the shared directory name will be removed. This should only be done if the shared directory was destroyed.

Note: This panel will not appear if the database is attached to an existing shared directory, you will get an error message on the previous selection panel instead.

Compuware DDIO Calculator — DDIOCALC

The Compuware DDIO Calculator is a tool for calculating your allocation and formatting of DDIO files (source and report) and databases (source, report and transaction). Although an existing file is not required to use this tool, it is highly recommended for use as a fine-tuning model. Otherwise, calculating file size and determining optimal formatting parameters will result in trial and error.

Note: It is not used for shared directory allocation or tuning.

Usage

The DDIO Calculator is available as a menu selection on the CSS Utilities main menu (Figure 3-1 on page 3-2); as a panel option on the CSS Utilities Source Listing File Facility screen (Figure 3-6 on page 3-11) and the Abend-AID Report File Facility screen (Figure 3-42 on page 3-37); or, optionally, as a TSO REXX command “DDIOCALC”, found in the CSS-supplied SLCXEXEC library.

Figure 3-74. DDIOCALC Primary Calculations Screen

```
            Compuware Shared Services DDIO File Size Calculator v1.1
                          Primary Calculations Screen

 Command ===>

  If you have a catalogued DDIO file, enter the DSN below
     === >
  Or, do you have DIRX report data Y/N     ==> Y

  Formatted Report Count (1 to 16380)  ==>
  Actual Number of Members             ==> N/A
  Average Member Size (in bytes)       ==>
  Largest Member in file (in bytes)    ==> N/A
  Blocksize or CISIZE (2048 to 32760)  ==>
   Blocks/Track: 2   Blocks/CYL: 30
  Group Count (1 to 16380)             ==>
  Extents (1 to 460)                   ==>

  Access Method (VSAM or SEQ)          ==> VSAM
  Device Type (3380 or 3390)           ==> 3390

  Total file size in cylinders         ==>

                    Press Enter to re-calculate, or END to exit.
```

An existing DDIO is recommended to check the file efficiency and to make adjustments. You can enter an existing DDIO file DSN name that will be used to obtain the necessary DIRX information and fill in the values. Alternately, you can enter the DIRX information manually. You will need to obtain a DIRX report for file statistics. This information can be obtained in one of two ways:

- Run a batch DIRX report (using CWDDSUTL for DDIO files, CWAASDUT for report databases, CWFXSDUT for transaction databases, or CWDDLPUT for source databases) against your current DDIO file or database.
- Select option “3. Source” or “4. AAReport” from the CSS Utilities Primary Menu. You will then place an 'I' next to the file of choice.

1. Enter the DSN for a catalogued DDIO file that can be used to obtain the DIRX data necessary (recommended).

   If a DSN is entered here, the DIRX information will be obtained directly from the file. The screen shown in Figure 3-74 will display with the values already filled in. Continue with step 4 on page 3-63.

2. Enter 'Y' or 'N' for an existing DIRX report (Optional).
3. Enter data obtained from the DIRX report.

If a ‘Y’ was entered for an existing DIRX report, the screen in Figure 3-75 will display:

Figure 3-75. DDIOCALC DIRX Report Data Input Screen

```
            Compuware Shared Services DDIO File Size Calculator v1.1
                         DIRX Report Data Input Screen

Enter the following information directly from the DIRX report:

  Access Method (VSAM or SEQ)         ==> VSAM
  Blksize                             ==> 0
  ReptCount                           ==> 0
  GroupCount                          ==> 0
  Extents                             ==> 0
  Blocks in File                      ==> 0
  Number of Members                   ==> 0
  Largest Member Size (in groups)     ==> 0
  Average Member Size (in groups)     ==> 0

Additional information:

  Device Type (3380 or 3390)          ==> 3390
  This DDIO file currently FULL? Y/N  ==> N

  Calculated file size in cylinders:

       Press Enter to re-calculate, or END to return to the primary screen.
```

The required fields can be taken directly from the DIRX report. Note that some of the information is located at the top of the DIRX report (Figure 3-76).

Figure 3-76. Top of the DIRX Report

```
ENTER UTILITY COMMAND:
  DIRX

DSN ----------------------------- PFHERSO.TRAIN.VSMREPT
VOLSER -------------------------- PRD950
AM(ACCESS METHOD) --------------- VSAM
BLKSIZE ------------------------- 4089
TYPE ---------------------------- ABEND-AID
REPTCOUNT ----------------------- 100
GROUPCOUNT ---------------------- 4
AUTODELETE ---------------------- YES
INDEX --------------------------- EXT
EXTENTS ------------------------- 460
OTHER INFORMATION
  BLOCKS IN FILE ---------------- 18000
  ALLOCATION GROUPS ------------- 4493
  USED ALLOCATION GROUPS -------- 80(2%)
  FILE FORMATTED BY DDIO RELEASE- 7.6.0
  REQUIRED MINIMUM DDIO RELEASE - 5.6.0
  CURRENT REPORT NUMBER --------- 3
  DIRECTORY ENTRIES/BLOCK ------- 4
  DIRECTORY BLOCKS -------------- 25
  FILE FORMAT ------------------- EXTENDED
  DDIO EXTENDED DIR SPACE ------- 6
  USER EXTENDED DIR SPACE ------- 0
```

The remaining information is located at the bottom of the DIRX report (Figure 3-77).

Figure 3-77. Bottom of the DIRX Report

```
MEMBER SUMMARY OF PFHERS0.TRAIN.VSMREPT

NO OF MEMBERS -------------------------------- 2
NO OF LOCKED MEMBERS ------------------------- 0
NO OF UNLOCKED MEMBERS ----------------------- 2
NO OF INCOMPLETE MEMBERS --------------------- 1
LARGEST MEMBER SIZE (IN GROUPS) -------------- 64
SMALLEST MEMBER SIZE (IN GROUPS) ------------- 14
AVERAGE MEMBER SIZE (IN GROUPS) -------------- 39.0

SPACE SUMMARY BASED ON AUTODELETE=YES
MAXIMUM ALLOCATION GROUPS -------------------- 4493
AVAILABLE ALLOCATION GROUPS ------------------ 4491
MAXIMUM NO OF DIRECTORY ENTRIES (REPTCOUNT) -- 100
AVAILABLE NO OF DIRECTORY ENTRIES ------------ 100

ENTER UTILITY COMMAND:
***  ALL FUNCTIONS SUCCESSFULLY COMPLETED
```

All required fields will highlight if left blank. After all fields have been entered, you will see the percentage of used space on the file:

Figure 3-78. DDIOCALC DIRX Report Data Input Screen

```
 Compuware Shared Services DDIO File Size Calculator v1.1
                        DIRX Report Data Input Screen

Enter the following information directly from the DIRX report:

 Access Method (VSAM or SEQ)         ==> VSAM
 Blksize                             ==> 4096
 ReptCount                           ==> 100
 GroupCount                          ==> 4            (Minimum = 1)
 Extents                             ==> 460
 Blocks in File                      ==> 18000
 Number of Members                   ==> 2
 Largest Member Size (in groups)     ==> 64           (1,048,576 BYTES)
 Average Member Size (in groups)     ==> 39           (638,976 BYTES)

Additional information:

 Device Type (3380 or 3390)          ==> 3390
 This DDIO file currently FULL? Y/N  ==> N            (1.7% Full)

 Calculated file size in cylinders:      100

      Press Enter to re-calculate, or END to return to the primary screen.
```

4. Make file allocation and format adjustments.

At this point, if the DIRX input screen is displayed, you may press PF3 to take you back to the first panel. Here, you can begin making adjustments to the file allocation by increasing/decreasing the file size and seeing the effects on the file, as well as making adjustments to the formatting parameters to increase DASD utilization.

Note: It is important to make one modification at a time and press Enter. It may be difficult to determine the effect on the file if you make multiple modifications simultaneously.

Helpful Hints

Here are a few suggestions to follow when making file adjustments.

- **Increase the number of listings that can be written to the file.**

Examine the file full percentage. Compare that value to the actual number of members. If the file is not 100% full, yet you have reached the maximum number of members, the file is not being utilized efficiently. The file will never run to 100% space capacity as it is limited by the report count set at the time it was formatted. Report Count defaults to 100. If the file is not full, and you have not reached the maximum RC, this calculator can be used to determine how many members, based on average size, can be written to the file.

If you want to see the effect on the file by increasing the number of actual members to a higher number, modify only one setting at a time based on the following scenarios:

– ***File not full, maximum report count not reached***

  Increase the actual number of members gradually until a file full condition is met. Was the maximum number of members reached? If not, are you satisfied with the number? If not, the file is too small and you will need to allocate more space. If the maximum number of members was reached before the file was full, see next item.

– ***File not full, maximum report count reached***

  First, increase the Report Count to an amount you feel will fill up the additional space or to a number you feel needs to be maintained to meet your shop's needs. Press Enter. There should be no visible change in the file percentages. If a substantial increase has been entered, there may be a warning about the required header record size.

  The next step is to increase the actual number of members gradually to meet a file full condition. This will tell you how many members will actual fit in the space provided.

  Note: Pay close attention to the total file size in cylinders as you make modifications to the actual number of members. It will automatically increase to accommodate the new number of members entered. If the new “number of members” value has exceeded the intended number of cylinders, decrease the number of members first, then decrease the total number of cylinders.

- **Increase the file space allocation to accommodate more members.**

  If the maximum number of members is not reached due to a file full condition, you may increase the number of cylinders to determine how much space is required to hold the desired number of members. You may also have a maximum file size and want to determine how many additional members will fit. This can be done in two ways:

  – ***Increase the file size based on required report count.***

    By increasing the report count and actual number of members to the desired amount, the total number of cylinders will automatically increase to accommodate the new total.

  – ***Increase the number of reports based on maximum file size.***

    The total number of cylinders can be increased to a desired amount. From there, the report count and actual number of members can be increased to determine how many members will actually fit within that file allocation.

- **Increase DASD utilization.**

  Ideally, DASD utilization should run near 100%. Any percentage under 90% is worth looking at to see if improvements can be made. The two parameters to consider for DASD utilization are blocksize (or CISIZE) and Group Count (GC). The physical allocation is divided into equal allocation groups (or Extents). The size of these Extents is determined by the blocking factor or CISIZE multiplied by the Group Count. A typical installation will have CISIZE=4096. If GC=4, each Extents setting

would be 4096 x 4 = 16384 bytes. The default Extents setting is 64. The Extents parameter is offered to override that default up to a maximum of 460.

Note: Keep in mind that the larger the allocation group is in size, the more potential waste there will be on the file. This means that if a member acquires an additional allocation group and only uses a few bytes, the remainder of the allocation group will not be used until the member has been deleted from the file. Multiply the possible amount of wasted space by the number of members in the file and you can see how a file may become full quickly without actually utilizing the space.

To increase DASD utilization, Compuware recommends third-track blocking. You accomplish this by allocating the file with CISIZE=18432. The previous practice of using CISIZE=4096 required Group Counts to be higher to accommodate large members. With the use of third-track blocking, a smaller GC will reduce the amount of wasted space per member. Third-track blocking requires the GC to be in multiples of 3.

Change the CISIZE first. You should note a change in DASD utilization immediately. Next, change the Group Count. You will not see DASD utilization change; however, you will notice a change in the maximum member size. This represents the size of the largest member this file can accommodate. Compare this to the actual largest member currently written to the file. If the actual member is much smaller, you may be able to lower the GC if it is higher than 3. Otherwise, the space is available to accommodate larger members, if needed in the future.

Chapter 4.
Batch File Utility CWDDSUTL

This chapter describes the batch file utility CWDDSUTL used by the following Compuware products:

- Abend-AID
- Abend-AID for CICS (source listing files only)
- XPEDITER/CICS
- XPEDITER/TSO

CWDDSUTL enables you to print, delete, lock, unlock, copy, move, import, and export DDIO file members, and to list DDIO file directory entries. It also enables you to initially format a DDIO file.

Note: Many CWDDSUTL functions may also be performed using Compuware's menu-driven CSS Utilities. For information about CSS Utilities, see **Chapter 3, "CSS Utilities"**.

For information on the CWFXSDUT utility for handling Abend-AID for CICS transaction databases and shared directories, refer to **Chapter 5, "Batch File Utility CWFXSDUT"**.

For information on the CWAASDUT utility for handling Abend-AID report databases and shared directories, refer to **Chapter 6, "Batch File Utility CWAASDUT"**.

For information on the CWDDLPUT utility for handling source listing databases and shared directories, refer to **Chapter 7, "Batch File Utility CWDDLPUT"**.

CAUTION:

CWDDSUTL is the only utility recommended for use with Compuware Source Listing DDIO files and Abend-AID Report DDIO files.

File-AID/MVS and other VSAM-related vendor products (including vendor packages that compress or enhance buffer management on VSAM files) should NOT be used against Compuware DDIO files. Doing so can corrupt the contents of the DDIO file.

Summary of CWDDSUTL Utility Commands

The following is a list and brief description of the commands available for the CWDDSUTL batch file utility. Each command is discussed in detail later in this chapter.

Command	Description
COPY	Copies members from one DDIO file to another DDIO file.
DELETE	Deletes members from a DDIO file.
DIRECTORY	Lists the directory entries in a DDIO file.
DIRS	Lists the directory entries in a sequential file produced by the EXPORT command.

Command	Description
DIRX	Lists the attributes of a DDIO file and its directory entries. In addition, this command lists the allocation groups for the members in a DDIO file.
EXPORT	Copies members from a DDIO file to a sequential file from which they can later be restored into the same or another DDIO file using the IMPORT command.
FLAG	Flags source lines in one or more members of a DDIO file to indicate which lines have changed since a specified previous version of the program.
FORMAT	Formats a newly allocated DDIO file, or reformats an existing sequential DDIO file.
IMPORT	Restores previously exported DDIO file members from a sequential dataset.
LOCK	Locks members of a DDIO file so that they cannot be automatically deleted when the DDIO file becomes full.
MOVE	Copies members between DDIO files and deletes them from the “from” file if they were successfully copied.
PRINT	Prints members of a DDIO file.
REORDER	Prepares a DDIO file originally formatted with a release of DDIO prior to release 5.7 (CSS 1.0) in order to perform an AUTODELETE based upon DATE/TIME rather than upon REPTNUMBER.
UNLOCK	Unlocks members of a DDIO file so that they can be automatically deleted when the DDIO file becomes full.

Usage Considerations

This section discusses special considerations for using the CWDDSUTL utility.

DD Statements Used with CWDDSUTL

The following is a description of each of the DD statements available for CWDDSUTL.

ABNLDFIL

The ABNLDFIL DD statement specifies the following:

- The input DDIO file for the DELETE, DIRECTORY, DIRS, DIRX, FORMAT, LOCK, PRINT, REORDER, and UNLOCK commands.
- The input sequential dataset containing exported members for the DIRS command.

The default ddname ABNLDFIL can be changed using the INDD parameter with any of these commands.

For the PRINT command, the ABNLDFIL DD statement specifies either a report file for printing Abend-AID diagnostic reports, or it specifies a source listing file for printing source listings for all products.

For example, to print a source listing:

```
//ABNLDFIL DD DSN=COMPWARE.LISTING.FILE,DISP=SHR
```

SLSF001

The SLSF001 DD statement specifies the source listing file for the PRINT command to provide source support for Abend-AID. In this case, the ABNLDFIL DD statement specifies the report file. If the SLSF001 DD statement is omitted, then the basic diagnostic report will be provided. The default ddname SLSF001 can be changed using the SLSDD parameter with the PRINT command.

The following examples illustrate the use of the ABNLDFIL and SLSF001 DD statements. These examples assume you are using the defaults for printing.

To print a diagnostic report without source support, or to print a source listing:

```
//ABNLDFIL DD DSN=COMPWARE.DDIO.FILE,DISP=SHR
```

To print a diagnostic report with source support:

```
//ABNLDFIL DD DSN=COMPWARE.REPORT.FILE,DISP=SHR
//SLSF001 DD DSN=COMPWARE.LISTING.FILE,DISP=SHR
```

Note: If more than one source listing file must be searched to provide source support for the PRINT command, the first source listing file must be specified by the xxxx001 DD statement, the second source listing file must be specified by the xxxx002 DD statement, etc. Up to eight source listing files can be used this way. By default, xxxx=SLSF. The default can be changed using the SLSDD parameter. For example, to use ddnames CICS001, CICS002, and so on up to CICS008, specify SLSDD=CICS001.

ABNLFROM

The ABNLFROM DD statement specifies the input DDIO file for the COPY, EXPORT, and MOVE commands, or it specifies the input sequential dataset for the IMPORT command.

The default ddname ABNLFROM can be changed using the FROMDD parameter.

The ABNLFROM DD statement is synonymous with the INDD parameter used in many DDSUTL commands. You can use either statement, but you cannot combine the two. If you attempt to use both statements, you will receive an error message.

ABNLTO

The ABNLTO DD statement specifies the output DDIO file for the COPY, IMPORT, and MOVE commands, or it specifies the output sequential dataset for the EXPORT command.

The default ddname ABNLTO can be changed using the TODD parameter.

ABNLTO DD Statement for the EXPORT Command

For the EXPORT command, the output sequential file is a fixed block file. Values for the LRECL and BLKSIZE parameters can be specified on the DCB parameter of the ABNLTO DD statement or can be taken from the DSCB of the existing DASD file.

The BLKSIZE cannot exceed the track capacity of the device to which the output sequential file is written. The BLKSIZE must be a multiple of the LRECL value. If the export is done to a tape, the BLKSIZE cannot exceed 32767 bytes.

If LRECL is not specified, it defaults to 80. If BLKSIZE is not specified, it defaults to 100 unless that blocksize exceeds the track capacity of the device. In that case, the largest blocksize that does not exceed the track capacity is used.

ABNLPARM

The ABNLPARM DD statement specifies the control dataset for the COPY, DELETE, DIRECTORY, DIRS, DIRX, EXPORT, FLAG, FORMAT, IMPORT, LOCK, MOVE, PRINT, REORDER, AND UNLOCK commands. Normally, the control dataset resides in the input stream. However, it can reside on a system input device or it can be defined as

a member in a partitioned dataset. You must specify the ABNLPARM DD statement each time you run the batch file utility. The ddname ABNLPARM cannot be altered.

ABNLREPT

The ABNLREPT DD statement specifies a sequential dataset for messages and reports produced by CWDDSUTL. It can also contain printed members for the PRINT command. You must specify the ABNLREPT DD statement each time you run the batch file utility. The ddname ABNLREPT cannot be altered.

anyname1

The dataset specified by the anyname1 DD statement must use RECFM=FBM and LRECL=133. The BLKSIZE must be a multiple of the LRECL value. If the dataset is a new dataset (DISP=NEW) and the DCB parameters are not specified, the default BLKSIZE value is 133.

Table 4-1 shows which DD statements can be used with each of the batch commands.

Table 4-1. DD Statements Used with CWDDSUTL Batch Commands

	ABNLDFIL	SLSF001	ABNLFROM	ABNLTO	ABNLPARM	ABNLREPT	anyname1
COPY			X	X	X	X	
DELETE	X				X	X	
DIRECTORY	X				X	X	
DIRX	X				X	X	
DIRS	X				X	X	
EXPORT			X	X	X	X	
FLAG					X	X	
FORMAT	X				X	X	
IMPORT			X	X	X	X	
LOCK	X				X	X	
MOVE			X	X	X	X	
PRINT	X	O			X	X	O
REORDER	X				X	X	
UNLOCK	X				X	X	
X — Required; **O** — Optional							

Usage Restrictions

1. When formatting DDIO files, the file being allocated cannot be compressed or altered by any other software products before CSS attempts to format the file.
2. CWDDSUTL is the only utility recommended for use with Compuware Source Listing DDIO files and Abend-AID Report DDIO files.
3. File-AID/MVS and other VSAM-related vendor products (including vendor packages that compress or enhance buffer management on VSAM files) should NOT be used against Compuware DDIO files. Doing so can corrupt the contents of the DDIO file.
4. You cannot COPY, MOVE, or IMPORT members from a DDIO file formatted with INDEX=EXT to a file formatted without that parameter. You can, however, COPY, MOVE, or IMPORT from a file formatted without the INDEX parameter to a file formatted with INDEX=EXT. INDEX=EXT is only useful for Abend-AID report files.

Command Conventions

Use the following conventions when specifying commands:

- Enter commands and their parameters as control cards in the dataset defined by the ABNLPARM DD statement.
- Commands can start in any column.
- Separate the command from its parameters with one or more spaces.
- Separate parameters with commas without intervening spaces.

```
EXPORT INC=NO,JOBNAME=ABCDEFG,ABEND=SOC4,DELETE=YES
```

- When typing parameter names, type either the whole parameter name, the shortcut in parentheses in the parameters list (see "Command Parameters for DDIO Files" on page 4-6), or the uppercase letters shown in the command syntax diagrams. Typing a shortened version of the parameter WILL NOT work. For example, typing PROG for the PROGRAM parameter is not valid — you must type PROGRAM or P. The same rules apply to typing command names.
- Control cards can also be continued on successive lines. Each line of a multiple-line control card statement must contain at least one parameter. To continue on a new line, enter a comma followed by a blank immediately after the last parameter on the line.

```
EXPORT INC=NO,JOBNAME=ABCDEFG,
ABEND=SOC4,DELETE=YES
```

- To include a comment on a line, enter at least one blank space after the last parameter on the line, followed by the comment text.

```
EXPORT INC=NO,JOBNAME=ABCDEFG, THIS IS A COMMENT
ABEND=SOC4,DELETE=YES          THIS IS ANOTHER COMMENT
```

- Any line containing an asterisk (*) followed by a space in the first two columns is treated as a comment line.
- The CWDDSUTL utility stops when it reaches an end-of-file (EOF) or null input control card.
- The asterisk (*) is the generic or wildcard character indicator, and can be used with the following parameters:
  - ABEND
  - JESID
  - JOBNAME
  - PROGRAM

  The wildcard character can only be used at the end of the parameter being specified.

  For example, to print all diagnostic reports for job names beginning with ABC for the given date range:

```
P JOBNAME=ABC*,STARTDATE=09/01/2002,ENDDATE=09/30/2002
```

  Note that while **ABC*** is a valid use of the wildcard character, **AB*C** is not valid.
- Parameter strings with blanks must have apostrophes around them.

Command Parameters for DDIO Files

This section describes the parameters accepted by the commands that manage DDIO files. Not all parameters are available for all commands.

ABEND (A):

Specifies Abend-AID diagnostic reports by abend code. Wildcard characters are valid. However, valid abend codes must begin with U or S.

AUTODELETE (AD):

Determines the action taken when an attempt is made to write a member to a Transaction Database file that is full. The AUTODELETE parameter specifies whether the oldest member in the DDIO file is automatically deleted when the file or directory becomes full. A DDIO file is considered full when all of the groups (EXTENTs) in the file have been allocated or the number of members specified by REPTCOUNT has been reached.

The autodelete process is based on **manual** and **automatic** locks. Manual locks are created online using the **L** line command from any directory screen or with the batch utility LOCK command. Automatic locks are created whenever a member is added to a DDIO file formatted using the AUTODELETE=DUPS or AUTODELETE=STAGED option. The most recent member is automatically locked.

Members that have been manually locked retain their locks during the autodeletion process. They may be unlocked by using the **U** line command online or with the UNLOCK batch command. A DELETE command in batch, or the **D** line command online, will unlock a member prior to executing the delete.

For source listings, the oldest member is the program with the earliest compile date and time. For diagnostic and transaction reports, the oldest member is the report with the earliest creation date and time. The following are valid values:

YES: (Default) Allows new members to be written to a full DDIO file by deleting the oldest unlocked members. This method is commonly referred to as **wrap-around** mode, and is typically used for DDIO report files and source listing files where members are added as needed. AUTODELETE=YES does not automatically lock members.

A file full condition will occur when AUTODELETE=YES and all members are manually locked or in use. You can resolve a file full condition by manually deleting members from the file or manually unlocking members.

Notes:

1. If you want to keep a specific member for a longer period, you can lock it, and it will be bypassed when autodelete processing selects the next report member to be automatically deleted.
2. The DELETE DUPS parameter should only be used against AUTODELETE=DUPS files. Using DELETE DUPS=UNLOCKED against an AUTODELETE=YES file will delete **all** unlocked members (‘ALL’ members).

DUPS: When a new member is added to a DDIO file, all automatically locked members with the same name are unlocked except the member with the most recent compile date and time (source listings) or abend date and time (diagnostic reports). Members with the same program name (source listings) or job name and abend code (Abend-AID reports) that have been manually locked using the **L** line command or the LOCK batch command remain locked and will not be deleted. If the DDIO file is full, the oldest unlocked member not in use, regardless of name, is chosen for autodeletion.

A file full condition will occur when AUTODELETE=DUPS and all members are manually or automatically locked or in use. You can resolve a file full condition by manually deleting members from the file or manually unlocking members.

STAGED: This option is similar to AUTODELETE=DUPS, and is valid for source listings only.

When a new member is added to a DDIO file, all automatically locked members with the same name are unlocked except the member with the most recent compile date and time. Members with the same name that have been manually locked remain locked. If the DDIO file is full, the oldest unlocked member not in use is chosen for autodeletion.

However, if all members are locked or in use, then the oldest automatically locked member not in use is deleted.

CAUTION:

If you need to retain the most current version of each source listing member, you should format your listing file with AUTODELETE=DUPS or AUTODELETE=NO.

A file full condition will occur when AUTODELETE=STAGED and all members are manually locked or in use. You can resolve a file full condition by manually deleting members from the file or manually unlocking members.

NO: Prevents members from being written to a DDIO file when it becomes full. To add members to a full file, one or more members of the file must be manually deleted using the DELETE command.

If the DDIO file holds production source listings, you may want to use AUTODELETE=NO so that the production listings are not automatically deleted before a new member is successfully added. A DDIO file that holds listings from development could be formatted with AUTODELETE=DUPS or AUTODELETE=STAGED. With either of these options, new members may be added to the full DDIO file if there are any members eligible to be automatically deleted.

A file full condition will occur when AUTODELETE=NO and the file is full. You can resolve a file full condition by manually deleting members from the file.

DDIO File Full Condition

A **file full condition** exists when new members cannot be added to a full DDIO file, in which case an error is returned. A DDIO file full condition will occur with the following AUTODELETE options:

YES: All members are manually or automatically locked or in use.

DUPS: All members are manually or automatically locked or in use.

STAGED: All members are manually locked or in use.

NO: The file is full.

Resolving a File Full Condition

You can perform the following steps to resolve a file full condition:

- Manually delete members from the file.
- Manually unlock members when AUTODELETE=YES, DUPS, or STAGED.
- Allocate and format a larger file, and use the MOVE or COPY commands to transfer the contents of the full file to the new file.
- Use the EXPORT command with the DELETE option to archive all or some members to a sequential file (tape or disk).

  Using EXPORT, space can be freed up without losing members, since members can be imported back into the file as needed. This method also avoids the overhead created by formatting new files.

BASEDDIO (BDDIO):

(Flag command only.) Specifies the ddname of the DDIO file that contains the baseline members. If the CURRDDIO file contains both the baseline and the current members, this DD statement is not required. The default is the ddname specified with the CURRDDIO parameter, which in turn defaults to "CURRENT."

BASENUMBER (BRN):

(FLAG command only.) Specifies a baseline member's listing number. This parameter is required if the PROGRAM parameter is not used.

BLKSIZE (BLK):

Specifies a blocksize for a sequential DDIO file. The default is 4096 (4 KB) and the maximum blocksize allowed is 32767. The BLKSIZE should be between 2048 (or 2 KB) and the track size.

If the BLKSIZE parameter is omitted for a sequential DDIO file, the file will be formatted with the default blocksize of 4096 (or 4 KB). The blocksize specified during file allocation is always ignored.

Note: For VSAM DDIO files, the blocksize is always equal to the VSAM Control Interval Size (CISIZE) minus 7 and cannot be overridden. The BLKSIZE parameter is ignored when formatting VSAM files.

CONTINUE:

Specifies whether the COPY, EXPORT, FLAG, IMPORT, or MOVE command will continue if recoverable errors are detected. A value of YES (default) specifies that the command will continue, if possible, when a recoverable error occurs. A value of NO specifies that the command will be stopped when an error occurs.

CURRDDIO (CDDIO):

(FLAG command only.) Specifies the ddname of the DDIO file that contains the current members. If this file also contains the baseline members, the BASEDDIO parameter should not be used. This parameter is required. The default value is the CURRENT DD statement.

CURRNUMBER (CRN):

(FLAG command only.) Specifies a current member's listing number. This parameter is required if the PROGRAM parameter is not used.

DAYSOLD:

Allows you to select members that are older than nnn days. For example, using the DELETE command with DAYSOLD=30 deletes all members that are more than 30 days old.

DELETE:

Specifies whether successfully exported members will be deleted from the "from" DDIO file. A value of YES specifies that the members will be deleted. A value of NO specifies that the members will not be deleted. The default value is NO.

DUPS:

Specifies whether all members, only locked members, or only unlocked members are selected. The available values are: ALL, LOCKED, or UNLOCKED. The default value is ALL.

ENDDATE (EDATE):

Specifies the last processing date of the diagnostic reports, or the last compile date of the source listings. This parameter can be specified as either MM/DD/YY or MM/DD/YYYY. Acceptable four-digit years are 1970 through 2069. If the ENDDATE parameter is specified, then the STARTDATE must also be specified. The highest valid ENDDATE is 12/31/2069.

ENDTIME (ETIME):

Specifies the last processing time of the diagnostic reports or the last compile time of the source listings. This parameter must be specified as HH:MM[:SS] or HH.MM[.SS]. If seconds are omitted, the default is 59.

This parameter must be preceded by the STARTTIME parameter. If the STARTTIME parameter is specified and the ENDTIME parameter is omitted, then ENDTIME defaults to 24:00:00.

When using STARTTIME and ENDTIME in conjunction with STARTDATE and ENDDATE, members are first selected by date. From that list, members are then selected based on the time range.

EXEC:

Specifies how the COPY, EXPORT, FLAG, IMPORT, and MOVE commands are executed.

Allowable values for COPY, EXPORT, IMPORT, and MOVE are:

SUMMARY: (Default) Transfers data and produces a summary report. The summary report contains a list of transferred members and a list of deleted "from" DDIO file members (if applicable).

CHECK: Produces a detailed report without any data being moved. In addition to the summary report, the detailed report includes the lists of attributes of the "to" and "from" DDIO files. For the MOVE, COPY, and IMPORT commands, it also includes the names of the members that will be locked and/or deleted in the "to" DDIO file, provided actual data movement occurs.

The detailed report produced by EXEC=CHECK may be different at RUN time if the "to" and/or "from" DDIO file is updated before the EXEC=RUN is performed.

Note: When EXEC=CHECK is specified, these commands are emulated with the header and directory blocks residing in memory. To keep the header and directory blocks in memory, you may need a larger region when executing these commands.

RUN: Transfers data and produces a detailed report. In addition to the summary report, the detailed report includes the lists of attributes of the "to" and "from" DDIO files. For the MOVE, COPY, and IMPORT commands, it also includes the names of locked and/or deleted members in the "to" DDIO file during execution of the command.

Allowable values for FLAG are:

RUN: Creates flagged members and produces a detailed report.

SIMULATE: Creates a detailed report, but does not create any flagged members.

EXTENTS:

Specifies the maximum number of extents to be allocated for each DDIO file member. If the EXTENTS parameter is not specified, the default value is 64. The maximum value is 460.

FLAGGEDDDIO (FDDIO):

(FLAG command only.) Specifies the ddname of a DDIO file to receive the tagged members. If the CURRDDIO file will be used for the tagged members, this parameter should not be specified. If you omit this parameter, each new flagged member will **replace** the corresponding member in the CURRDDIO file. If you supply this parameter, the baseline and current members will not be altered or replaced. Instead, a new flagged member will be created in the FLAGGEDDDIO file.

FROMDD:

Specifies the ddname of the DDIO file from which the members are copied, exported, or moved. It also specifies the ddname of the sequential file from which previously exported members are imported. When the FROMDD parameter is not specified, the default ddname ABNLFROM is used.

GROUPCOUNT (GC):

Specifies the number of blocks allocated to a DDIO file member each time additional space is needed. The minimum value is 1, and the maximum value is 2048. The default value is 4.

INC:

Specifies whether to include incomplete members in the selection. Allowable values are:

YES: (Default) Select both complete and incomplete members.

NO: Select only complete members.

ONLY: Select only incomplete members.

ID:

Specifies the user-defined identifier and its location. The first parameter is the ID, which has a maximum length of eight bytes. The ID must be contained within columns 73-80. The second parameter is the starting column position. The values are enclosed in parentheses. For example: ID=(BOB,73). This field will be used as a filter by the Code Coverage reporting facility.

INDD (I):

Specifies an alternative ddname for the DDIO file. This parameter also specifies the ddname of a sequential file containing exported members for the DIRS command. If the INDD parameter is not specified, the default ddname ABNLDFIL is used.

INDEX:

When used, this parameter's only permissible value of EXT allows indexes to occupy more than one block. This parameter is only useful for Abend-AID report files.

CAUTION:

You cannot COPY, MOVE, or IMPORT members from a DDIO file formatted with INDEX=EXT to a file formatted without that parameter. You can, however, COPY, MOVE, or IMPORT from a file formatted without the INDEX parameter to a file formatted with INDEX=EXT. INDEX=EXT is only useful for Abend-AID **report files.**

Note: The maximum size for DDIO files is 4 gigabytes, which is not the same as the maximum size for VSAM files.

JESID:

For Abend-AID, specifies diagnostic reports by JESID.

JOBNAME (J):

For Abend-AID, this parameter specifies diagnostic reports by job name. The value of the parameter is an eight-character job name. Wildcard characters are valid.

```
JOBNAME=ABC*   or   J=ABC*
```

LABEL:

Specifies the file number for the sequential dataset on the tape for the IMPORT, EXPORT, and DIRS commands. (VOL=SER=, and other parameters should be specified on the corresponding DD statement). The default value is the value coded in the JCL. If a value is not specified in the JCL, then the default value is 1.

The LABEL parameter enables you to reference multiple export files using a single ddname. This provides a means by which you can create multiple export files on a single tape. for example, to include a DDIO file on a tape that already contains the related source code and compiler output as files 1 and 2, you can specify LABEL=3 in the EXPORT command for that DDIO file. This will append the DDIO file on the tape as file 3. To perform a DIRS or IMPORT on that file at a later time, include the LABEL parameter in your JCL.

The LABEL parameter will only be recognized for IMPORT, EXPORT, and DIRS.

Note: If you include the LABEL parameter in your JCL, but have also specified a LABEL parameter with the IMPORT, EXPORT, or DIRS batch command, the LABEL parameter in the JCL will be overwritten.

For compatibility with the IBM Tape Management System, the label number ".Lnnn" is appended to the tape DSNAME as the last qualifier. If the combined length of the specified DSNAME and the appended label number exceeds the maximum length for DSNAME, an error message is issued.

LANGUAGE (LANG):

Specifies the available language support and the type of letter case used. Valid values are:

ENGLISH: (Default) Mixed-case English

USAMIX: Alias for mixed-case English

USAUC: Uppercase English

LINESIZE (LIN):

Specifies the line width of the PRINT command output. Allowable values are:

S: (Default) Short, 80-column output

L: Long, 133-column output

The LINESIZE parameter is valid only for report files. Source listing files always use 133 columns.

MONTHSOLD:

Allows you to select members that are older than nnn months. For example, using the EXPORT command with MONTHSOLD=6 exports all members that are older than six months.

NOTFOUND:

Specifies whether to request a return code 00, instead of the default code 44, when a member is not found. The valid values are:

ERROR: (Default) Allows the default code.

IGNORE: Forces a return code 00.

OUTDD (O):

Specifies the ddname for the alternative output file for the PRINT command. If the OUTDD parameter is not specified, the output of the PRINT command is written to the file specified by the ABNLREPT DD statement, along with all CWDDSUTL messages and reports.

See anyname1 on page 4-4 for more information on this ddname and the PRINT command.

PAGESIZE (PG):

Specifies the number of lines per page for the PRINT command output. Allowable values are any number from 20 to 99. The default value is 60.

PROGRAM (P):

Specifies source listings by program name. This parameter accepts program names of up to eight characters. Wildcard characters are valid.

```
PROGRAM=PAYROLL
```

REPTCOUNT (RC):

Specifies the maximum number of members in a DDIO file, provided there is enough space. The minimum value is 1, and the maximum value is 16330. The default is 100.

REPTNUMBER (RN):

Specifies a member's number. If this parameter is specified, all other selective parameters are ignored. Maximum length is ten digits.

SECONDARY:

Allows VSAM DDIO files to be allocated with secondary extents. This allows a VSAM DDIO file to be allocated across multiple volumes.

Notes:

a. This parameter is only applicable for VSAM DDIO files, and the VSAM cluster must be pre-allocated.

b. VSAM DDIO files do not operate in the same way as regular VSAM files pertaining to secondary extents.

SEEFLAGS:

(PRINT command only.) Specifies whether text added by a previous run of the FLAG command should be included in the printed line. Valid values are:

YES: (Default) Text added by the FLAG command is included in the output.

NO: Text added by the FLAG command is suppressed.

SLSDD:

For Abend-AID source support, this parameter specifies the ddname of the optional source listing file for which the PRINT command will provide source support. If the SLSDD parameter is not specified, the default ddname SLSF001 is used.

If the SLSDD parameter is omitted and the SLSF001 DD statement is not coded, only the basic diagnostic report will be provided by the PRINT command.

SORT:

Specifies the order in which members are selected for the execution of a command.

By default, members are selected in descending date/time order:

- Members are first ordered by date, from newest (most recent) to oldest.
- Members with the same date are then ordered by time (from latest to earliest).

The various types of DDIO files, valid sort keys, their abbreviations, and the sorting sequence they cause are specified in Table 4-2 and Table 4-3.

Table 4-2. Abend-AID Report File Sort Keys

Sort Key	Abbreviation	Primary Sort Field	Sequence	Secondary Sort Field	Sequence	Tertiary Sort Field	Sequence
JOBNAME	JN	JOBNAME	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
JESID		JESID	Descending	DATE/TIME	Descending	REPT-NUMBER	Descending
DESCRIPTION	DESC	DESCRIPTION	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
SIZE		SIZE	Descending	DATE/TIME	Descending	REPT-NUMBER	Descending
NUMBER	NUM	REPT- NUMBER	Descending				
CODE		CODE	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
DATE		DATE/TIME	Descending	REPT-NUMBER	Descending		

Table 4-3. Source Listing File Sort keys

Sort Key	Abbreviation	Primary Sort Field	Sequence	Secondary Sort Field	Sequence	Tertiary Sort Field	Sequence
PROGRAM	PGM	PROGRAM	Ascending	DATE/TIME	Descending	LIST-NUMBER	Descending
NUMBER	NUM	LIST-NUMBER	Descending				
DATE		DATE/TIME	Descending	LIST-NUMBER	Descending		
RC (return code)		RC	Ascending	DATE/TIME	Descending	LIST-NUMBER	Descending
LANGUAGE	LANG	LANGUAGE	Ascending	DATE/TIME	Descending	LIST-NUMBER	Descending
SIZE		SIZE	Descending	DATE/TIME	Descending	LIST-NUMBER	Descending

STARTDATE (SDATE):

Specifies the processing date of the diagnostic reports, or the compile date of the source listings. The STARTDATE parameter is used with the ENDDATE parameter to specify a date range.

The STARTDATE parameter can be specified as either MM/DD/YY or MM/DD/YYYY. Acceptable four-digit years are 1970 through 2069. When this parameter is used

alone, all members for the specified date are selected. The earliest valid start date is 01/01/1970.

STARTTIME (STIME):

Specifies the beginning processing time of the diagnostic reports, or the beginning compile time of the source listings. This parameter must be specified as HH:MM[:SS] or HH.MM[.SS]. If seconds are omitted, the default is 00. This parameter must precede the ENDTIME parameter (if ENDTIME is specified).

If the STARTTIME parameter is specified and the ENDTIME parameter is omitted, then ENDTIME defaults to 24:00:00.

When using STARTTIME and ENDTIME in conjunction with STARTDATE and ENDDATE, members are first selected by date. From that list, members are then selected based on the time range.

SUBTYPE:

Further specifies the type of members the DDIO file can contain. This parameter is used in conjunction with the TYPE parameter.

- If you are creating source listing files for XPEDITER/CICS release 6.0, 6.01, or 6.03, you must use this parameter. Code TYPE=CICS,SUBTYPE=SOURCE.
- If you are formatting source listing files for Abend-AID for CICS and you are using user or system DSECT files, you must specify SUBTYPE=FX. The TYPE parameter defaults to TYPE=CICS.

TODD:

Specifies the ddname of the DDIO file to which the members are copied, imported, or moved. It also specifies the ddname of the sequential file to which members are exported. If the TODD parameter is not specified, the default ddname ABNLTO is used.

TYPE:

Specify the kind of members the DDIO file can contain.

- For Abend-AID report files, omit the TYPE parameter.
- For source listing files, code TYPE=SOURCE unless you are using a version of XPEDITER/CICS prior to 6.2. If you are using XPEDITER/CICS release 6.0, 6.01, or 6.03, you must code TYPE=CICS,SUBTYPE=SOURCE.
- If you are using a security exit, and the exit tests for the value of the TYPE parameter, you may be required to use TYPE=CICS,SUBTYPE=SOURCE.

YEARSOLD:

Allows you to select members that are older than nnn years. For example, using the UNLOCK command with YEARSOLD=1 unlocks all members that are older than 1 year.

Command Descriptions and Syntax

The following is a detailed description of the commands available for existing DDIO files, including valid command abbreviations, and the parameters allowed for each command.

COPY

The COPY command copies selected members from one DDIO file to another DDIO file. Use an ABNLFROM DD statement to specify the "from" file, and use the ABNLTO DD statement to specify the "to" file. If these ddnames are not used, the FROMDD and TODD parameters must specify the "from" and "to" file ddnames.

If no other parameters are specified, then all members are copied.

```
>>-COPY-+-------------------------+,-+-----------------------+,-->
        +-ReptNumber=xxxxxxxxxx---+   +-Program=xxxxxxx*-------+
                                      +-Jobname=xxxxxxx*--------+

>--+-----------------+,--+-----------------+,--+-----------------+,-->
   +-JESID=xxxxxxx*--+   +-Abend=xxxx*-------+   +-SORT=sortkey----+

>--+-----------------------------------------------------------+,-->
   +-DAYSOLD=nnn-------------------------------------------------+
   +-MONTHSOLD=nnn-----------------------------------------------+
   +-YEARSOLD=nnn------------------------------------------------+
   +-StartDATE=mm/dd/yyyy,---+-----------------+-----------------+
                             +-EndDATE=mm/dd/yyyy--+

>--+-----------------------------------------------------------+,-->
   +-StartTIME=hh:mm[:ss],---+-EndTIME=24:00:00-+-------------+
                             +-EndTIME=hh:mm[:ss]-+-------------+

>--+-DUPS=ALL-------+,--+-INC=YES-----+,--+-FROMDD=ABNLFROM-+,--+-TODD=ABNLTO-+,-->
   +-DUPS=LOCKED----+   +-INC=NO------+   +-FROMDD=ddname----+   +-TODD=ddname-+
   +-DUPS=UNLOCKED--+   +-INC=ONLY----+

>--+-CONTINUE=YES-+,--+-NOTFOUND=ERROR-+,--+-EXEC=SUMMARY-+-><
   +-CONTINUE=NO--+   +-NOTFOUND=IGNORE-+   +-EXEC=CHECK---+
                                            +-EXEC=RUN-----+
```

For a description of the parameters, see "Command Parameters for DDIO Files" on page 4-6.

Example

Copy all locked source listings for the programs compiled in March 2002 between 7:30 AM and 4:30 PM.

```
COPY SDATE=03/01/2002,EDATE=03/31/2002,STIME=07:30,ETIME=16:30,DUPS=LOCKED
```

DELETE (DEL)

The DELETE command deletes selected members from the DDIO file specified by the ABNLDFIL DD statement. If this ddname is not used, the INDD parameter must specify the DDIO file ddname.

At least one parameter must be entered.

```
►►─DELete─┬──────────────────────┬─,─┬──────────────────────┬─,──►
          └─ReptNumber=xxxxxxxxxx─┘   ├─Program=xxxxxxx*──┤
                                      └─Jobname=xxxxxxx*──┘

►─┬──────────────┬─,─┬──────────┬─,──────►
  └─JESID=xxxxxxx*─┘   └─Abend=xxxx*─┘

►─┬──────────────────────────────────────────────────────┬─,──►
  ├─DAYSOLD=nnn──────────────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,─┬──────────────────────┬──────┘
                          └─EndDATE=mm/dd/yyyy─┘

►─┬──────────────────────────────────────────────────────┬─,──►
  │                          ┌─EndTIME=24:00:00──┐        │
  └─StartTIME=hh:mm[:ss],──┬─────────────────────┬──────┘
                           └─EndTIME=hh:mm[:ss]──┘

  ┌─DUPS=ALL──────┐   ┌─INC=YES──┐   ┌─Indd=ABNLDFIL─┐
►─┼───────────────┼─,─┼──────────┼─,─┼───────────────┼─,──►
  ├─DUPS=LOCKED───┤   ├─INC=NO───┤   └─Indd=ddname───┘
  └─DUPS=UNLOCKED─┘   └─INC=ONLY─┘

  ┌─NOTFOUND=ERROR──┐
►─┼─────────────────┼──►◄
  └─NOTFOUND=IGNORE─┘
```

For a description of the parameters, see “Command Parameters for DDIO Files” on page 4-6.

Example

Delete all incomplete members in the DDIO file that are older than six months.

```
DELETE INC=ONLY,MONTHSOLD=6
```

DIRECTORY (DIR, D) and DIRX (DX)

The DIRECTORY command lists selected directory entries in the DDIO file specified by the ABNLDFIL DD statement. If this ddname is not used, the INDD parameter must specify the DDIO file ddname.

The DIRX command functions similar to the DIRECTORY command, but it also displays the attribute and allocation information.

If no other parameters are specified, then all entries are listed.

```
►►─DIRectory or DIRX─┬──────────────────────┬─┬──────────────────────┬─┬──►
                     └─ReptNumber=xxxxxxxxxx─┘ ├─Program=xxxxxxx*──┤
                                               └─Jobname=xxxxxxx*──┘

►─┬──────────────┬─┬──────────────┬─┬──►
  └─JESID=xxxxxxx*─┘ └─Abend=xxxx*─┘

►─┬──────────────────────────────────────────────────────────┬─┬──►
  ├─DAYSOLD=nnn──────────────────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,─┬──────────────────────┬──────────┘
                          └─EndDATE=mm/dd/yyyy─┘

►─┬──────────────────────────────────────────────────────────┬─┬──►
  │                                 ┌─EndTIME=24:00:00──┐    │
  └─StartTIME=hh:mm[:ss],─┬─────────────────────────┬──┘
                          └─EndTIME=hh:mm[:ss]──┘

  ┌─DUPS=ALL──────┐  ┌─INC=YES──┐  ┌─NOTFOUND=ERROR──┐
►─┼───────────────┼─┼───────────┼─┼─────────────────┼─┬──►
  ├─DUPS=LOCKED───┤  ├─INC=NO───┤  └─NOTFOUND=IGNORE──┘
  └─DUPS=UNLOCKED─┘  └─INC=ONLY─┘

                     ┌─Indd=ABNLDFIL─┐
►─┬──────────────┬─┬──────────────┬──►◄
  └─SORT=sortkey─┘ └─Indd=ddname───┘
```

For a description of the parameters, see “Command Parameters for DDIO Files” on page 4-6.

Example

Print a DIRX of all programs beginning with the prefix COPZ:

```
DIRX PROGRAM=COPZ*,INDD=SRC1
```

DIRS

The DIRS (DIRectory Sequential) command lists selected directory entries in a sequential file produced by the EXPORT command and specified by the ABNLDFIL DD statement. Use the INDD parameter to change this ddname.

If no other parameters are specified, then all entries are listed.

```
►►─DIRS─┬──────────────────────┬─┬──────────────────────┬─┬──►
        └─ReptNumber=xxxxxxxxxx─┘ ├─Program=xxxxxxx*──┤
                                  └─Jobname=xxxxxxx*──┘

►─┬──────────────┬─┬──────────────┬─┬──────────►
  └─JESID=xxxxxxx*─┘ └─Abend=xxxx*─┘

►─┬──────────────────────────────────────────────────────┬─┬──►
  ├─DAYSOLD=nnn──────────────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,──┬──────────────────────────┬─┘
                           └─EndDATE=mm/dd/yyyy───────┘

►─┬──────────────────────────────────────────────────────┬─┬──►
  │                                                      │
  └─StartTIME=hh:mm[:ss],──┬─EndTIME=24:00:00──────┬──┘
                           ├──────────────────────┤
                           └─EndTIME=hh:mm[:ss]───┘

  ┌─DUPS=ALL──────┐  ┌─INC=YES──┐  ┌─NOTFOUND=ERROR──┐
►─┼───────────────┼─┼──────────┼─┼─────────────────┼─┼──►
  ├─DUPS=LOCKED───┤  ├─INC=NO───┤  └─NOTFOUND=IGNORE──┘
  └─DUPS=UNLOCKED─┘  └─INC=ONLY─┘

                   ┌─Indd=ABNLDFIL─┐  ┌─LABEL=1──┐
►─┬──────────────┬─┼───────────────┼─┼──────────┼──►◄
  └─SORT=sortkey─┘ └─Indd=ddname───┘  └─LABEL=nn───┘
```

For a description of the parameters, see “Command Parameters for DDIO Files” on page 4-6.

Example

Export all unlocked members and list all directory entries on the resulting sequential file.

```
EXPORT DUPS=UNLOCKED
DIRS INDD=ABNLTO
```

EXPORT

The EXPORT command exports selected members from the DDIO file specified by the ABNLFROM DD statement to a sequential file specified by the ABNLTO DD statement. If these ddnames are not used, the FROMDD and TODD parameters must specify the "from" and "to" file ddnames. If no other parameters are specified, then all members are exported.

You may use multiple EXPORT commands if you include a unique TODD parameter for each output dataset or PDS member that is a target of the export. If the TODD parameter is not used, each successive export will overlay the previous export. Refer to the LABEL parameter under "YEARSOLD:" on page 4-14 for additional information on multiple export files and referencing those files using the LABEL parameter.

```
>>-EXPORT-+-------------------------+,-+-------------------------+,-------->
          +-ReptNumber=xxxxxxxxxx-+         +-Program=xxxxxxx*-+
                                            +-Jobname=xxxxxxx*-+

>-+-------------+,-+-------------------+,-------->
  +-Abend=xxxx*-+   +-JESID=xxxxxxx*-+

>-+-------------------------------------------------------------+,-->
  +-DAYSOLD=nnn-------------------------------------------------+
  +-MONTHSOLD=nnn-----------------------------------------------+
  +-YEARSOLD=nnn------------------------------------------------+
  +-StartDATE=mm/dd/yyyy,---------------------------------------+
                                +-EndDATE=mm/dd/yyyy-+

>-+-------------------------------------------------------------+,-->
  +                                 +-EndTIME=24:00:00-+
  +-StartTIME=hh:mm[:ss],-----------+-------------------+
                                    +-EndTIME=hh:mm[:ss]-+

  +-DUPS=ALL-------+   +-INC=YES-+   +-NOTFOUND=ERROR-+
>-+-------------+,-+-------------+,-+-------------+,-->
  +-DUPS=LOCKED--+   +-INC=NO--+   +-NOTFOUND=IGNORE-+
  +-DUPS=UNLOCKED-+   +-INC=ONLY-+

  +-FROMDD=ABNLFROM-+   +-TODD=ABNLTO-+   +-CONTINUE=YES-+
>-+-------------+,-+-------------+,-+-------------+,-->
  +-FROMDD=ddname--+   +-TODD=ddname-+   +-CONTINUE=NO--+

  +-LABEL=1-+   +-EXEC=SUMMARY-+   +-DELETE=NO-+
>-+-------+,-+-------------+,-+-------------+-->
  +-LABEL=nn-+   +-EXEC=CHECK--+   +-DELETE=YES-+
                 +-EXEC=RUN----+
```

For a description of the parameters, see "Command Parameters for DDIO Files" on page 4-6. Refer to Shared Services SLXCNTL installation library member CXEXPORT for sample JCL for the EXPORT command.

Example

For an Abend-AID report file, export all complete diagnostic reports for the jobs with names that start with ABCD and abended with system abend codes 0C1 through 0CF. Delete all successfully exported members.

```
EXPORT INC=NO,JOBNAME=ABCD*,ABEND=SOC*,DELETE=YES
```

FLAG

FLAG modifies or creates a source listing (member in a DDIO file) to show which of its source lines have changed since a prior version of the program. Data definition source lines as well as executable source lines are compared and flagged as appropriate.

The "old" or "baseline" version of the program is contained in the DDIO file specified with the BASEDDIO parameter. The "current" version of the program is contained in the DDIO file specified with the CURRDDIO parameter. FLAG compares these two versions of the program and creates a new version that looks like the "current" version **except** that lines inserted or modified since the "baseline" version are flagged. Flags consist of a user-specified string of characters placed at a user-specified location on the source line (within the range of columns 73-80), as denoted by the ID parameter.

Note: There is no indication of lines present in the "baseline" version but subsequently deleted from the "current" version.

The new "flagged" member can be created in a separate DDIO file, with no alteration of either "baseline" or "current" members, or it can be created in the same DDIO file as the "current" member, in which case it **replaces** the "current" member. The "baseline" member is not altered, replaced, or deleted in either case. The FLAGGEDDDIO parameter is used to specify the separate DDIO file to receive the flagged members. If this parameter is omitted, the new flagged members replace the corresponding "current" members in the CURRDDIO file.

There are three ways to specify that DDIO members represent the "baseline" and the "current" versions of the program:

1. Use the BASENUMBER= and CURRNUMBER= parameters to specify (by member number) which two members are to be compared. The BASENUMBER= value is the member number of the "baseline" version of the program in the DDIO file specified in the BASEDDIO parameter. The CURRNUMBER= value is the member number of the "current" version of the program in the DDIO file specified in the CURRDDIO parameter.
2. Use the PROGRAM= parameter to specify the name of a program. FLAG locates members in the BASEDDIO and CURRDDIO files with this name, and selects the BASEDDIO member with the oldest compile date and time as the "baseline" member, and the CURRDDIO member with the most recent compile date and time as the "current" member.

   Note: Members from different language families (Assembler, COBOL, PL/I, or C) are not compared, even if their program names match.

3. Use the PROGRAM= parameter to specify a mask for a program name. The mask is 0-7 characters of a program name followed by an asterisk; for example, "PROGRAM=ACC*" or "PROGRAM=*". FLAG uses this mask to identify matching program names, then proceeds as described in step 2.

FLAG locates all pairs of programs (one in BASEDDIO and one in CURRDDIO) of the same language family and with the same name, with a common name matching the mask. FLAG selects the program in BASEDDIO with the oldest compile date and time as the "baseline" member of the pair, and the one in CURRDDIO with the most recent compile date and time as the "current" member of the pair. FLAG then compares these two, and creates a new flagged member in the FLAGGEDDDIO file (or replaces the "current" member, if the FLAGGEDDDIO parameter was not specified).

Repeated Use of the FLAG Command

To ensure the most efficient use of I/O and CPU resources, FLAG does not "reflag" currently flagged members on subsequent runs when the FLAGGEDDDIO parameter is omitted.

For example, imagine that you have used FLAG to compare two members, and have omitted the FLAGGEDDDIO parameter to ensure that lines changed from the baseline to the current member are flagged in the current DDIO file. If you run FLAG a second time on the same two members, CWDDSUTL detects the changes that have already been flagged, and bypasses these listings. This logic permits you to run FLAG against entire DDIO files, and use only the resources needed to analyze the changed programs in these files.

FLAG Report Output

FLAG printed output contains the following:

- The dataset names of the BASEDDIO, CURRDDIO, and (if used) FLAGGEDDDIO datasets.
- The name, member number, etc., of the "baseline," "current," and (if FLAGGEDDDIO is used) flagged members.

If a program mask is used, there are potentially multiple pairs of members to display in the report. In this case, the report contains two sections that list members:

1. A "Selection Report" showing every pair of listings that **will be** compared. This list is created early in FLAG processing to ensure that it is available in the job output, even if the job is subsequently canceled.
2. An "Action Report" showing the actions taken with each pair of members in the Selection Report.

If you run with EXEC=SIMULATE, **only** the Selection Report is produced. This report can then be used to verify and approve all proposed comparisons. If necessary, you can use individual lines from the Selection Report to create separate FLAG commands that specify BASENUMBER= and CURRNUMBER= parameters in lieu of a PROGRAM= mask. Multiple FLAG commands can be entered in a single run of the CWDDSUTL utility.

Supported Compilers

FLAG supports the following languages:

- IBM COBOL for OS/390 & VM 2.1 and 2.1.1 (MLE)
- IBM COBOL for MVS & VM 1.2 and 1.2.2 (MLE)
- IBM COBOL II 1.2, 1.3, 1.4
- IBM OVS/VS COBOL 2.4
- IBM PL/I for MVS & VM
- IBM OS PL/I Optimizing Compiler 2.2, 2.3
- IBM PL/I Optimizing Compiler 1.5.1
- IBM Assembler H version 2
- IBM High Level Assembler 1.1, 1.2, 1.3
- C/MVS V3 R2
- OS/390 C releases through V2 R10.

Note: CA-OPTIMIZER products are not supported.

Flag Command Syntax

The following diagram illustrates the syntax for the FLAG command.

```
                 ┌CurrDDIO=CURRENT─┐
►►─FLAG──────────┴CurrDDIO=ddname──┴─┬──────────────────┬─┬──►
                                     └BaseDDIO=ddname───┘

►─┬──────────────────────┬─┬──────────────────┬─┬──►
  └FlaggedDDIO=ddname────┘ └Program=xxxxxxx*──┘

►─┬──────────────────┬─┬──────────────────┬─┬──►
  └BASENUMBER=nnnn───┘ └CuRrNumber=nnn────┘

                                  ┌CONTINUE=YES─┐
►─ID=(string,column-number),──────┼─────────────┼─┬──►
                                  └CONTINUE=NO──┘

  ┌EXEC=RUN───────┐
►─┼───────────────┼──►◄
  └EXEC=SIMULATE──┘
```

For a description of the parameters, see “Command Parameters for DDIO Files” on page 4-6.

Example

Flag the newest version of the programs with names beginning with “PREP” to show which lines differ from the oldest version. Place the string “ARQ” in columns 76-78 of the altered/new lines.

```
FLAG PROGRAM=PREP*,ID=(ARQ,76)
```

FORMAT Command

The FORMAT command prepares a DDIO file for use after the file has been allocated, or it reformats an existing sequential DDIO file.

Note: See **Chapter 2, "Allocating and Formatting DDIO Files"** for more information about using these parameters and certain restrictions when setting up DDIO files.

CAUTION:
The FORMAT command erases all of the information in the specified file.

The file being formatted is specified with the ABNLDFIL DD statement. The INDD parameter may be used to override the ddname.

- For Abend-AID report files, all parameters are optional. Code a parameter only if a default will be changed.
- For source listing files, specify TYPE=SOURCE unless you are using a version of XPEDITER/CICS prior to 6.2. If you are using XPEDITER/CICS release 6.0, 6.01, or 6.03, you must specify TYPE=CICS,SUBTYPE=SOURCE. All other parameters are optional.
- For Abend-AID for CICS source listing files, when you are formatting system or user DSECT files, code SUBTYPE=FX. TYPE=CICS is the default when SUBTYPE=FX is specified. All other parameters are optional.

To reformat a VSAM DDIO file, you must first delete the file, and then reallocate it before formatting it again. If the size of the VSAM DDIO file must be increased, do not use IDCAMS and the REPRODUCE command to copy the file. Instead, allocate a new file, format the file, and use the COPY or MOVE commands to copy or move members from the old DDIO file to the new DDIO file. If you specify the SPEED parameter as part of the DEFINE CLUSTER command, the FORMAT command will execute faster.

Do not use the FORMAT command from a CMS terminal; you may encounter problems due to enqueue and dequeue restrictions.

FORMAT Command Sample Programs

Sample programs for the FORMAT command are available in CSS installation library members. Use sample JCL member CXFMTDS and refer to Table 4-4 for a list of the sample parameter members that contain information on using the FORMAT command.

Table 4-4. Sample Programs for FORMAT Command

To format...	Use sample parameter member...
Report Files	CXAARPT
Source Listing files	CXFMTLST

FORMAT Command Syntax

The following diagram illustrates the syntax for the FORMAT command.

```
                                                   ┌ReptCount=100──┐
►► FORMAT──┬──────────────────────────────────┬──┼───────────────┼──►
           ├TYPE=SOURCE──────────────────────┤  └ReptCount=nnnnn─┘
           └TYPE=CICS,──┬──────────────────┬─┘
                        ├SUBTYPE=SOURCE────┤
                        └SUBTYPE=FX────────┘

   ┌GroupCount=4──┐      ┌Indd=ABNLDFIL──┐
►──┼──────────────┼──┬──┼───────────────┼──┬──►
   └GroupCount=nnnn─┘  └Indd=ddname────┘

   ┌AutoDelete=YES──────┐      ┌BLKsize=4096──┐
►──┼────────────────────┼──┬──┼──────────────┼──┬──►
   ├AutoDelete=DUPS─────┤  └BLKsize=nnnnn─┘
   ├AutoDelete=STAGED───┤
   └AutoDelete=NO───────┘

   ┌SECONDARY=NO──┐      ┌EXTENTS=64──┐
►──┼──────────────┼──┬──┼────────────┼──┬──┬──────────┬──┬──►◄
   └SECONDARY=YES─┘  └EXTENTS=nnn─┘  └INDEX=EXT─┘
```

For a description of the parameters, see “Command Parameters for DDIO Files” on page 4-6.

Notes:

1. The SECONDARY parameter is only valid for VSAM DDIO files that have been pre-allocated.
2. This parameter is only useful for Abend-AID report files.

Example 1:

Format a sequential Abend-AID report file with a blocksize of 8 KB, and allow extended indexes to be used.

```
FORMAT BLKSIZE=8192,INDEX=EXT
```

Example 2:

Format a source listing file to be used by Abend-AID and XPEDITER/CICS release 6.2.

```
FORMAT TYPE=SOURCE,GROUPCOUNT=8,AUTODELETE=DUPS
```

Example 3:

Format a source listing file to be used by Abend-AID and a version of XPEDITER/CICS prior to 6.2. The file should be capable of containing up to 150 source listings.

```
FORMAT TYPE=CICS,SUBTYPE=SOURCE,GROUPCOUNT=8,AUTODELETE=DUPS,RC=150
```

IMPORT

The IMPORT command selectively restores previously exported members from the sequential file specified by the ABNLFROM DD statement into the DDIO file specified by the ABNLTO DD statement. If these ddnames are not used, the FROMDD and TODD parameters must specify the “from” and “to” file ddnames.

Note: If the access method is **VSAM**, then you must specify **DISP=SHR** for the **TODSN** DD statement in your JCL.

If no other parameters are specified, then all exported members are imported.

Multiple IMPORT commands can be used, provided that you include a unique TODD parameter for each input dataset or PDS member that is a target of the import. If the TODD parameter is not used, each successive import will overlay the previous import. Refer to the LABEL parameter under “YEARSOLD:” on page 4-14 for additional information on multiple import files and referencing those files using the LABEL parameter.

```
>>-IMPORT-+-------------------------+,-+-------------------+,-+-------------------+,->
          +-ReptNumber=xxxxxxxxxx-+     +-Program=xxxxxxx*-+     +-JESID=xxxxxxx*-+
                                        +-Jobname=xxxxxxx*-+

>-+-------------+,-+-------------------------------------------------------+,->
  +-Abend=xxxx*-+   +-DAYSOLD=nnn-------------------------------------------+
                    +-MONTHSOLD=nnn-----------------------------------------+
                    +-YEARSOLD=nnn------------------------------------------+
                    +-StartDATE=mm/dd/yyyy,-+-------------------------------+
                                            +-EndDATE=mm/dd/yyyy-+

>-+-------------------------------------------------------------+,->
  +-StartTIME=hh:mm[:ss],-+-EndTIME=24:00:00-+-+
                          +-EndTIME=hh:mm[:ss]-+

  +-DUPS=ALL-------+,-+-INC=YES-+,-+-FROMDD=ABNLFROM-+,-+-TODD=ABNLTO-+,->
>-+-------------+,-+-------+,-+-------------+,-+-------------+,->
  +-DUPS=LOCKED---+   +-INC=NO--+   +-FROMDD=ddname--+   +-TODD=ddname-+
  +-DUPS=UNLOCKED-+   +-INC=ONLY-+

  +-CONTINUE=YES-+,-+-NOTFOUND=ERROR-+,-+-LABEL=1-+,-+-EXEC=SUMMARY-+-><
>-+-------------+,-+-------------+,-+-------+,-+-------------+-><
  +-CONTINUE=NO--+   +-NOTFOUND=IGNORE-+   +-LABEL=nn-+   +-EXEC=CHECK---+
                                                        +-EXEC=RUN-----+
```

For a description of the parameters, see “Command Parameters for DDIO Files” on page 4-6. Refer to Shared Services SLXCNTL installation library member CXIMPORT for sample JCL for the IMPORT command.

Example

Restore all members from the fifth sequential file on the tape.

```
IMPORT LABEL=5
```

LOCK

The LOCK command manually locks selected members of the DDIO file specified by the ABNLDFIL DD statement. If this ddname is not used, code the INDD parameter to specify the DDIO file ddname.

While a member is manually locked, it cannot be automatically deleted when the DDIO file becomes full.

Note: Manually locked members are designated with an (M) on the DIR/DIRX report and on the directory screen of Compuware/VF.

At least one parameter must be specified.

```
►►─LOCK─┬──────────────────────┬─,──┬──────────────────────┬─,──►
        └ReptNumber=xxxxxxxxxx─┘    ├Program=xxxxxxx*──┤
                                    └Jobname=xxxxxxx*──┘

►─┬──────────────┬─,──┬──────────────┬─,──►
  └JESID=xxxxxxx*─┘    └Abend=xxxx*─┘

►─┬──────────────────────────────────────────────────────┬─,──►
  ├DAYSOLD=nnn──────────────────────────────────────────┤
  ├MONTHSOLD=nnn────────────────────────────────────────┤
  ├YEARSOLD=nnn─────────────────────────────────────────┤
  └StartDATE=mm/dd/yyyy,──┬──────────────────────┬──┘
                          └EndDATE=mm/dd/yyyy─┘

►─┬──────────────────────────────────────────────────────┬─,──►
  │                          ┌EndTIME=24:00:00──┐        │
  └StartTIME=hh:mm[:ss],──┼──────────────────────┼──┘
                             └EndTIME=hh:mm[:ss]──┘

  ┌DUPS=ALL──────┐   ┌INC=YES──┐   ┌NOTFOUND=ERROR──┐
►─┼──────────────┼─,─┼─────────┼─,─┼──────────────────┼─,──►
  ├DUPS=LOCKED───┤   ├INC=NO───┤   └NOTFOUND=IGNORE──┘
  └DUPS=UNLOCKED─┘   └INC=ONLY─┘

  ┌Indd=ABNLDFIL─┐
►─┼──────────────┼──────────►◄
  └Indd=ddname───┘
```

For a description of the parameters, see “Command Parameters for DDIO Files” on page 4-6.

Example

Lock diagnostic report number 777.

```
LOCK REPTNUMBER=777
```

MOVE

The MOVE command copies selected members from one DDIO file to another DDIO file and deletes them from the “from” DDIO file, provided they are successfully copied. Use the ABNLFROM DD statement to specify the “from” file, and use the ABNLTO DD statement to specify the “to” file. If these ddnames are not used, the FROMDD and TODD parameters must specify the “from” and “to” file ddnames.

If no other parameters are specified, then all members are moved.

Note: Only successfully copied members are deleted from the “from” DDIO file.

```
>>-MOVE-+-------------------------+,-+-------------------------+,->
        +-ReptNumber=xxxxxxxxxx-+     +-Program=xxxxxxx*-------+
                                      +-Jobname=xxxxxxx*-------+

>-+-------------------+,-+-------------+,-+-------------------+,->
  +-JESID=xxxxxxx*-+     +-Abend=xxxx*-+     +-SORT=sortkey-+

>-+-------------------------------------------------------------+,->
  +-DAYSOLD=nnn-------------------------------------------------+
  +-MONTHSOLD=nnn-----------------------------------------------+
  +-YEARSOLD=nnn------------------------------------------------+
  +-StartDATE=mm/dd/yyyy,-----+-------------------------------+
                              +-EndDATE=mm/dd/yyyy-+

>-+-------------------------------------------------------------+,->
  +-StartTIME=hh:mm[:ss],-----+-EndTIME=24:00:00-----+-----+
                              +-EndTIME=hh:mm[:ss]-+

  +-DUPS=ALL-------+   +-INC=YES-----+   +-FROMDD=ABNLFROM-+
>-+-------------------+,-+-------------+,-+-------------------+,->
  +-DUPS=LOCKED----+   +-INC=NO------+   +-FROMDD=ddname----+
  +-DUPS=UNLOCKED--+   +-INC=ONLY----+

  +-TODD=ABNLTO-+   +-CONTINUE=YES-+   +-NOTFOUND=ERROR-----+
>-+-------------+,-+-------------+,-+-------------------+,->
  +-TODD=ddname-+   +-CONTINUE=NO--+   +-NOTFOUND=IGNORE----+

  +-EXEC=SUMMARY-+
>-+--------------+-->
  +-EXEC=CHECK---+
  +-EXEC=RUN-----+
```

For a description of the parameters, see “Command Parameters for DDIO Files” on page 4-6.

Example

Move diagnostic report number 1 to the ABNLTO file.

```
MOVE RN=1
```

PRINT (P)

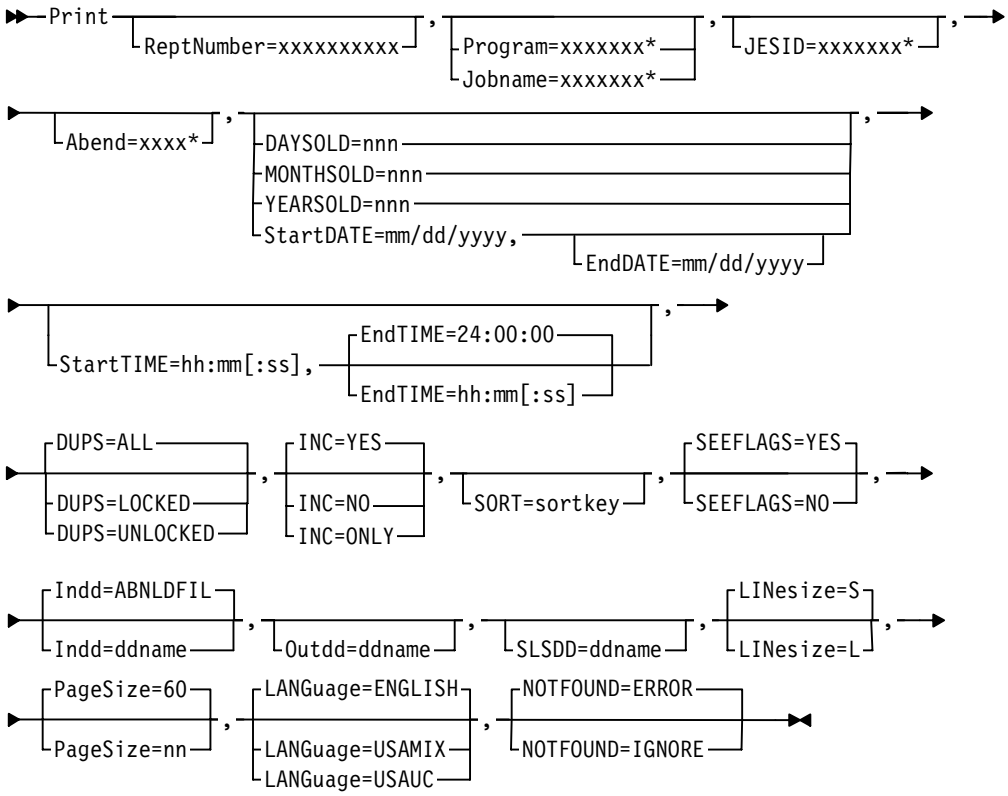
The PRINT command writes selected members of the DDIO file specified by the ABNLDFIL DD statement. If this ddname is not used, the INDD parameter must specify the DDIO file ddname.

For Abend-AID report processing, both the CSS and product load libraries are required in the STEPLIB.

To provide source support, the source listing file must be specified by the SLSF001 DD statement or the DD statement specified by the SLSDD parameter. See the note in the "SLSF001" statement under "Usage Considerations" on page 4-2 if you require more than one source listing file.

If the OUTDD parameter is omitted, the output (along with other CWDDSUTL messages and reports) is sent to the file specified by the ABNLREPT DD statement. If the OUTDD parameter is coded, the output is sent to the file with the ddname specified by the OUTDD parameter.

At least one parameter must be specified.



For a description of the parameters, see "Command Parameters for DDIO Files" on page 4-6. See the Shared Services SLXCNTL installation library member CXPDDSU for sample JCL to run the PRINT command.

Example

Print report number 207.

```
PRINT RN=207
```

REORDER

The REORDER command rearranges DDIO files so that the AUTODELETE feature is based on date/time rather than on REPTNUMBER.

Initially, you should run the REORDER command once against all DDIO files formatted with a release of DDIO prior to 5.7. After that, use REORDER to rearrange source listing files formatted with DDIO release 5.7 or later whenever any members are inadvertently added using a release of DDIO prior to 5.7. The REORDER command can be executed as often as necessary.

The ABNLDFIL DD statement specifies the DDIO file to be converted. Use the INDD parameter if this ddname must be changed.

```
                 ┌Indd=ABNLDFIL─┐
▶▶─-REORDER──────┼──────────────┼──────────▶◀
                 └Indd=ddname───┘
```

For a description of the parameters, see “Command Parameters for DDIO Files” on page 4-6.

Example

Rearrange the DDIO file formatted with DDIO release 5.6.

```
REORDER
```

UNLOCK

The UNLOCK command unlocks selected members of the DDIO file specified by the ABNLDFIL DD statement. If this ddname is not used, code the INDD parameter to specify the DDIO file ddname.

When a member is unlocked, it can be deleted automatically when the DDIO file becomes full if the DDIO file has been formatted with AUTODELETE=YES, AUTODELETE=STAGED, or AUTODELETE=DUPS.

At least one parameter must be specified.

```
►►─UNLOCK─┬──────────────────────────┬─,─┬──────────────────────┬─,──►
          └─ReptNumber=xxxxxxxxxx─┘   ├─Program=xxxxxxx*─┤
                                      └─Jobname=xxxxxxx*─┘

►─┬──────────────┬─,─┬──────────┬─,──────►
  └─JESID=xxxxxxx*─┘   └─Abend=xxxx*─┘

►─┬─────────────────────────────────────────────────────┬─,──►
  ├─DAYSOLD=nnn──────────────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,─┬──────────────────────┬──────┘
                          └─EndDATE=mm/dd/yyyy─┘

►─┬─────────────────────────────────────────────────────┬─,──►
  │                                  ┌─EndTIME=24:00:00─┐  │
  └─StartTIME=hh:mm[:ss],────────────┼──────────────────┼──┘
                                     └─EndTIME=hh:mm[:ss]─┘

  ┌─DUPS=ALL─────┐   ┌─INC=YES─┐   ┌─Indd=ABNLDFIL─┐
►─┼──────────────┼─,─┼─────────┼─,─┼───────────────┼─,──►
  ├─DUPS=LOCKED──┤   ├─INC=NO──┤   └─Indd=ddname───┘
  └─DUPS=UNLOCKED─┘   └─INC=ONLY─┘

  ┌─NOTFOUND=ERROR──┐
►─┼─────────────────┼──►◄
  └─NOTFOUND=IGNORE─┘
```

For a description of the parameters, see “Command Parameters for DDIO Files” on page 4-6.

Example

Unlock all members in the report file that are older than 1 year.

```
UNLOCK JOBNAME=*,YEARSOLD=1
```

Chapter 5.
Batch File Utility CWFXSDUT

This chapter describes the batch file utility CWFXSDUT used by Abend-AID for CICS.

CAUTIONS:

> **To run this utility, the Abend-AID for CICS (SKFXLOAD) and Base Services (SKMPLOAD) non-authorized load libraries must be available in the link list or in the STEPLIB or JOBLIB in your JCL.**
>
> **CWFXSDUT is the only utility recommended for use with Compuware** Abend-AID for CICS **Shared Directories and Transaction Databases.**
>
> **CWFXSDUT must be used to delete shared directories and/or Transaction Databases to avoid possible internal corruption.**
>
> **File-AID/MVS and other VSAM-related vendor products (including vendor packages that compress or enhance buffer management on VSAM files) should NOT be used against Compuware DDIO files. Doing so can corrupt the contents of the DDIO file.**

CWFXSDUT allows you to manage the files used by a shared directory and its associated transaction databases. The shared directory is used to process all directory requests related to a transaction database.

For information on the CWDDSUTL utility for handling source listing files and Abend-AID report files, refer to **Chapter 4, "Batch File Utility CWDDSUTL"**.

For information on the CWAASDUT utility for handling Abend-AID report databases and shared directories, refer to **Chapter 6, "Batch File Utility CWAASDUT"**.

For information on the CWDDLPUT utility for handling source listing databases and shared directories, refer to **Chapter 7, "Batch File Utility CWDDLPUT"**.

The term **entry** is used throughout this chapter to refer to a transaction diagnostic report in a transaction database.

Summary of CWFXSDUT Utility Commands

The following is a list and brief description of the commands available for the CWFXSDUT batch file utility. Each command is discussed in detail later in this chapter.

ATTACH	Connects a standalone transaction database to a shared directory and populates the shared directory with transaction database entries.
CHANGE	Changes the AUTODELETE attribute of a transaction database.
COPY	Copies entries from one shared directory or transaction database to another shared directory or transaction database.
CREATE	Allocates and formats a shared directory or transaction database, or formats a pre-allocated shared directory or transaction database.
DELETE	Deletes a selected shared directory or transaction database or selected transaction database entries.

DETACH Disconnects a transaction database from a shared directory and removes the transaction database entries from the shared directory.

DIRECTORY Lists selected entries in a shared directory or transaction database.

DIRS Lists selected directory entries in a sequential file produced by the EXPORT command.

DIRX Reports on the characteristics of a shared directory and all transaction databases in the shared directory. Reports on the characteristics of the entries in a transaction database.

ESTIMATE Calculates the number of blocks to allocate for a shared directory when the allocation is being performed outside of CSS (i.e., through IDCAMS or Compuware's File-AID product).

EXAMINE Reads, analyzes, and optionally repairs the selected shared directory. Optionally deletes incomplete members in the shared directory and attached databases (if DELETEIC is specified).

EXPAND Resizes a shared directory's space allocation and/or CISIZE.

EXPORT Exports selected entries from a shared directory or transaction database to a sequential file, from which they can later be imported into the same or another shared directory or transaction database using the IMPORT command.

IMPORT Selectively restores previously exported entries from a sequential file to a shared directory or transaction database.

LOCK Locks the entries of a selected transaction database, or the entries of all transaction databases attached to a specified shared directory, so entries cannot be automatically deleted when the transaction database becomes full.

MOVE Copies entries from one shared directory or transaction database to another shared directory or transaction database and deletes them from the "from" file only if they were successfully copied.

RESET Disconnects a transaction database from a shared directory when the shared directory is lost or damaged. The transaction database can then be attached to another shared directory.

SCRATCH Clears all entries from a shared directory for a specified transaction database that has been lost or damaged.

UNLOCK Unlocks selected entries of a specified transaction database, or all transaction databases attached to a specified shared directory, so that they can be automatically deleted when the transaction database becomes full.

VERIFY Ensures that the information contained in a shared directory matches the information contained in its transaction databases. This command produces a report listing any inconsistencies but does not perform any maintenance or repairs.

Usage Considerations

This section discusses special considerations for using the CWFXSDUT utility.

Renaming Shared Directories and Transaction Databases

To rename a transaction database, you must first detach it from the shared directory and then rename it. You must then re-attach it to the shared directory in order to retain all of the correct references. The transaction database, with its new name, must be connected to the shared directory before it can be used.

To rename a shared directory, you must first detach the transaction databases from the old shared directory. Next, re-attach the transaction databases to the renamed shared directory.

DD Statements Used with CWFXSDUT

The following is a description of each of the DD statements required for CWFXSDUT.

Table 5-1 contains a list of the required and optional DD statements used with CWFXSDUT batch commands.

ABNLPARM

The ABNLPARM DD statement specifies the control dataset. Normally, the control dataset resides in the input stream; however, it can reside on a system input device or it can be defined as a member in a partitioned dataset. You must specify the ABNLPARM DD statement each time you run the batch file utility. The ddname ABNLPARM cannot be altered.

ABNLREPT

The ABNLREPT DD statement specifies a sequential dataset for messages and reports produced by CWFXSDUT. You must specify the ABNLREPT DD statement each time you run the batch file utility. The ddname ABNLREPT cannot be altered.

ABNLFROM

The ABNLFROM DD statement specifies the input transaction database for the COPY, EXPORT, IMPORT, and MOVE commands. The default ddname ABNLFROM can be changed using the FROMDD parameter.

ABNLTO

The ABNLTO DD statement specifies the output transaction database for the COPY, EXPORT, IMPORT, and MOVE commands. The default ddname ABNLTO can be changed using the TODD parameter.

FDBDCUST

The FDBDCUST DD statement points to the Abend-AID for CICS customization file. It is used to allocate the shared directory and transaction database and is required for Abend-AID for CICS. For more information on the customization file, refer to the *Abend-AID for CICS Installation and Customization Guide*.

Table 5-1. DD Statements Used with CWFXSDUT Batch Commands

	ABNLPARM	ABNLREPT	ABNLFROM	ABNLTO	FDBDCUST
ATTACH	X	X			X
CHANGE	X	X			X
COPY	X	X	O	O	X
CREATE	X	X			X

Table 5-1. DD Statements Used with CWFXSDUT Batch Commands

	ABNLPARM	ABNLREPT	ABNLFROM	ABNLTO	FDBDCUST
DELETE	X	X			X
DETACH	X	X			X
DIR	X	X			X
DIRS	X	X			X
DIRX	X	X			X
ESTIMATE	X	X			X
EXAMINE	X	X			X
EXPAND	X	X			X
EXPORT	X	X	O	O	X
IMPORT	X	X	O	O	X
LOCK	X	X			X
MOVE	X	X	O	O	X
RESET	X	X			X
SCRATCH	X	X			X
UNLOCK	X	X			X
VERIFY	X	X			X
X — Required; O — Optional					

Usage Restrictions

1. To run CWFXSDUT, the Abend-AID for CICS (SKFXLOAD) and Base Services (SKMPLOAD) non-authorized load libraries must be available in the link list or in the STEPLIB or JOBLIB in your JCL.
2. CWFXSDUT is the only utility recommended for use with Compuware Abend-AID for CICS Shared Directories and Transaction databases.
3. CWFXSDUT must be used to delete shared directories and/or transaction databases to avoid possible internal corruption.
4. File-AID/MVS and other VSAM-related vendor products (including vendor packages that compress or enhance buffer management on VSAM files) should NOT be used against Compuware DDIO files. Doing so can corrupt the contents of the DDIO file.

Command Conventions

Use the following conventions when specifying commands:

- Enter commands and their parameters as control cards in the dataset defined by the ABNLPARM DD statement.
- Commands can start in any column.
- Separate the command from its parameters by one or more spaces.
- Separate parameters by commas without intervening spaces.
- When typing parameter names, type either the whole parameter name, the shortcut in parentheses in the parameters list (see "Command Parameters for Shared Directories and Transaction Databases" on page 5-5), or the uppercase letters shown in the command syntax diagram. Typing a shortened version of the parameter WILL NOT work. For example, typing AUTOD for the AUTODELETE parameter is not valid — you must type AUTODELETE or AD. The same rules apply to typing command names.
- Control cards can also be continued on successive lines. Each line of a multiple-line control card statement must contain at least one parameter. To continue on a new line, enter a comma followed by a space immediately after the last parameter on the line.
- To include a comment on a line, enter at least one blank space after the last parameter on the line, followed by the comment text.
- Any line containing an asterisk (*) followed by a space in the first two columns is treated as a comment line.
- The CWFXSDUT utility stops when it reaches an end-of-file (EOF) or null input control card.
- The asterisk (*) is the generic or wildcard character indicator, and it can be used in the following parameters:
  - APPLID
  - CICSCODE
  - TRAN

  The wildcard character can only be used at the end of the parameter being specified.
- Parameter strings with blanks must have apostrophes around them.

Command Parameters for Shared Directories and Transaction Databases

This section provides information on the command parameters for shared directories and transaction databases. Not all parameters are available for all commands.

AM

Specifies the access method for the file being created. This parameter is used only when allocating or reallocating a file. Allowable values are:

VSAM: The access method is VSAM.

BSAM: The access method is sequential.

Note: A shared directory must be created as a VSAM file.

APPLID

Specifies transactions by system APPLID. Wildcard characters are valid.

AUTODELETE (AD):

Determines the action taken when an attempt is made to write a member to a Transaction Database that is full. The AUTODELETE parameter specifies whether the oldest member in the DDIO file is automatically deleted when the file or directory becomes full. A DDIO file is considered full when all of the groups (EXTENTs) in the file have been allocated or the number of members specified by REPTCOUNT has been reached.

The autodelete process is based on **manual** and **automatic** locks. Manual locks are created online using the **L** line command from any directory screen or with the batch utility LOCK command. Automatic locks are created whenever a member is added to a DDIO file formatted using the AUTODELETE=DUPS option. The most recent member is automatically locked.

Members that have been manually locked retain their locks during the autodeletion process. They may be unlocked by using the **U** line command online or with the UNLOCK batch command. A DELETE command in batch, or the **D** line command online, will unlock a member prior to executing the delete.

For transaction reports, the oldest member is the report with the earliest creation date and time. The following are valid values.

YES: (Default): Allows new members to be written to a full DDIO file by deleting the oldest unlocked members. This method is commonly referred to as **wrap-around** mode, and is typically used for DDIO report files and source listing files where members are added as needed. AUTODELETE=YES does not automatically lock members.

A file full condition will occur when AUTODELETE=YES and all members are manually locked or in use. You can resolve a file full condition by manually deleting members from the file or manually unlocking members.

Notes:

1. If you want to keep a specific member for a longer period, you can lock it, and it will be bypassed when autodelete processing selects the next report member to be automatically deleted.
2. The DELETE DUPS parameter should only be used against AUTODELETE=DUPS files. Using DELETE DUPS=UNLOCKED against an AUTODELETE=YES file will delete all unlocked members (‘ALL’ members).

DUPS: When a new member is added to a DDIO file, all automatically locked members with the same name are unlocked except the member with the most recent abend date and time (transaction reports). Members with the same transaction and abend code that have been manually locked using the **L** line command or the LOCK batch command remain locked and will not be deleted. If the DDIO file is full, the oldest unlocked member not in use, regardless of name, is chosen for autodeletion.

A file full condition will occur when AUTODELETE=DUPS and all members are manually or automatically locked or in use. You can resolve a file full condition by manually deleting members from the file or manually unlocking members when AUTODELETE=DUPS.

NO: Prevents members from being written to a DDIO file when it becomes full. To add members to a full file, one or more members of the file must be manually deleted using the DELETE command.

A file full condition will occur when AUTODELETE=NO and the file is full. You can resolve a file full condition by manually deleting members from the file.

DDIO File Full Condition

A **file full condition** exists when new members cannot be added to a full DDIO file, in which case an error is returned. A DDIO file full condition will occur with the following AUTODELETE options:

YES All members are manually or automatically locked or in use.

DUPS All members are manually or automatically locked or in use.

NO The file is full.

Resolving a File Full Condition

You can perform the following steps to resolve a file full condition:

– Manually delete members from the file.

– Manually unlock members when AUTODELETE=YES or AUTODELETE=DUPS.

– Allocate and format a larger file, and use the MOVE or COPY commands to transfer the contents of the full file to the new file.

– Use the EXPORT command with the DELETE option to archive all or some members to a sequential file (tape or disk).

Using EXPORT, space can be freed up without losing members, since members can be imported back into the file as needed. This method also avoids the overhead created by formatting new files.

BLKSIZE (BLK)

If you specified AM=BSAM, use this parameter to specify a blocksize for the sequential transaction database. The default is 4096 (or 4 KB). The BLKSIZE should be between 2048 (or 2 KB) and the track size. The maximum BLKSIZE is 32767.

CICSCODE (CC)

Specifies transaction reports by abend code. Wildcard characters are valid.

CISIZE

If you have specified AM=VSAM, use this parameter to specify the control interval size.

If neither the CISIZE nor the BLKSIZE parameter is specified, then the default CISIZE is 4096, and the default blocksize is 4089.

If the CISIZE parameter is not specified and the BLKSIZE parameter is specified, then the value for CISIZE is determined by adding 7 to the BLKSIZE value.

If the BLKSIZE parameter is not specified and the CISIZE parameter is specified, then the blocksize is determined by subtracting 7 from the CISIZE value.

If both the CISIZE and the BLKSIZE parameters are specified, then the BLKSIZE value is ignored. In this case, the blocksize is determined by subtracting 7 from the CISIZE value.

CONTINUE

Specifies whether the COPY, EXPORT, IMPORT, or MOVE command will continue if recoverable errors are detected. A value of YES (default) specifies that the command

will continue, if possible, when a recoverable error occurs. A value of NO specifies that the command will be stopped when an error occurs.

CYLINDERS (CYL)

To use cylinders as the unit of measure for the size of the file, use this parameter to specify the number of cylinders.

DATABASEDISP or DBDISP

When a shared directory is deleted, this parameter determines the disposition of the transaction databases attached to the shared directory. Allowable values are:

DETACH: (Default) Detach all attached transaction databases from the shared directory prior to deleting the shared directory. This choice preserves the transaction databases and allows you to attach them to another shared directory.

DELETE: Do not detach the transaction databases from the shared directory. This choice deletes both the shared directory and the attached transaction databases.

DATACLAS

Specifies the eight-character data class name for the Storage Management Subsystem (SMS) operating environment. This parameter is only valid for new shared directories or transaction databases, or when the REALLOCATE parameter is set to YES.

DAYSOLD

Allows you to select entries in a shared directory or transaction database that are older than nnn days. For example, using the DELETE command for a transaction database with DAYSOLD=30 deletes all transaction report entries that are more than 30 days old.

DDNAME (DDN)

If you have specified OBJECT=DATABASE, use this parameter for the ddname of the JCL statement that specifies the dataset name of the transaction database.

DELETE

Specifies whether successfully exported entries will be deleted from the “from” transaction database. A value of YES specifies that the entries will be deleted. A value of NO specifies that the entries will not be deleted. The default value is NO.

DELETEIC

Deletes any incomplete reports from the shared directory and databases. Incomplete reports are partial dump reports that did not complete because of CICS region outages, and/or Abend-AID for CICS dump capture region abends, or capture timeout (ABTIME exceeded). This parameter is valid for the EXAMINE command.

DIRDD

If you have specified OBJECT=SHRDIR, this parameter indicates the ddname of the shared directory.

If you have specified OBJECT=DATABASE, this parameter indicates the ddname of the shared directory to which the transaction database will be attached.

DIRDSN

If you have specified OBJECT=SHRDIR, and you are **not** using a JCL statement for the dataset name, use this parameter to specify the dataset name of the shared directory. This parameter contains the actual dataset name.

If you have specified OBJECT=DATABASE, and you are **not** using a JCL statement for the dataset name of the shared directory, you can use this parameter to specify the

dataset name of the shared directory to which the transaction database will be attached.

Note: You can use either the **DIRDD** or the **DIRDSN** parameter, depending on whether you are using a JCL statement for the dataset name of the shared directory.

DIRENTS

If you have specified OBJECT=SHRDIR, and you are unable to determine the number of blocks you will need for the shared directory, you can use this parameter to allocate an appropriate number of blocks. This parameter specifies the total number of directory entries expected in the shared directory. The total number of directory entries consists of the total number of transaction reports for each transaction database attached to the shared directory plus the total number of region dumps. When combined with a blocksize, the DIRENTS parameter is used by CWFXSDUT to calculate a reasonable number of blocks for allocation. The blocksize can be specified directly in the BLKSIZE parameter or indirectly through the CISIZE or MODEL parameter.

If you are using an external method to allocate blocks (e.g., IDCAMS or Compuware's File-AID), you can use the ESTIMATE command to determine the number of blocks you will need. See "ESTIMATE" on page 5-27 for more information.

DIRVOL

If you have specified OBJECT=SHRDIR, and you are **not** using a JCL statement for the volume serial number of the shared directory, you can use this parameter to specify the volume serial number.

DSN

If you have specified OBJECT=DATABASE, and you are **not** using a JCL statement for the dataset name, use this parameter to specify the dataset name of the transaction database.

DUPS

Specifies whether all entries, only locked entries, or only unlocked entries are selected. The available values are: ALL, LOCKED, or UNLOCKED. The default value is ALL.

ENDDATE (EDATE)

Specifies the last processing date of the transaction reports. This parameter can be specified as either MM/DD/YY or MM/DD/YYYY. Acceptable four-digit years are 1970 through 2069. If ENDDATE is specified, then STARTDATE must also be specified. The highest valid ENDDATE is 12/31/2069.

ENDTIME (ETIME)

Specifies the last processing time of the transaction reports. This parameter must be specified as HH:MM[:SS] or HH.MM[.SS]. If seconds are omitted, the default is 59.

This parameter must be preceded by the STARTTIME parameter. If the STARTTIME parameter is specified and the ENDTIME parameter is omitted, then ENDTIME defaults to 24:00:00.

When using STARTTIME and ENDTIME in conjunction with STARTDATE and ENDDATE, entries are first selected by date, then by time.

EXTENTS

Specifies the maximum number of extents to be allocated for each transaction database entry. If the EXTENTS parameter is not specified, the default value is 64. The maximum value is 460.

FROMDD

Specifies the ddname of the transaction database or shared directory from which the entries are copied, exported, or moved. It also specifies the ddname of the sequential file from which previously exported entries are imported.

FROMDSN

Specifies the DSN of the transaction database or shared directory from which the entries are copied, exported, or moved. It also specifies the DSN of the sequential file from which previously exported entries are imported.

FROMOBJECT

Specifies the type of file from which the entries are copied, exported, or moved. Allowable values are:

SHRDIR: The "from" file is a shared directory.

DATABASE: The "from" file is a transaction database.

FROMVOL

Specifies the VOLSER of the transaction database or shared directory from which the entries are copied, exported, or moved. It also specifies the VOLSER of the sequential file from which previously exported entries are imported.

GROUPCOUNT (GC)

Specifies the number of blocks allocated to a transaction database entry each time additional space is needed. The minimum value is 1, and the maximum value is 2048. The default value is 4.

INC

Specifies whether to include incomplete entries in the selection. Allowable values are:

YES: (Default) Select both complete and incomplete entries.

NO: Select only complete entries.

ONLY: Select only incomplete entries.

LABEL

Specifies the file number for the sequential dataset on the tape for the IMPORT, EXPORT, and DIRS commands. (VOL=SER=, and other parameters should be specified on the corresponding DD statement). The default value is 1.

The LABEL parameter allows you to reference multiple export files using a single ddname. It also allows you to create multiple export files on a single tape. For example, to include a transaction database on a tape that already contains the related source code and compiler output as files 1 and 2, you can specify LABEL=3 in the EXPORT command for the transaction database. This will append the transaction database on the tape as file 3. Later, to perform a DIRS or IMPORT on that file, include LABEL=3 with the command.

The LABEL parameter is valid for IMPORT, EXPORT, and DIRS.

Note: If you include the LABEL parameter in your JCL, but have also specified a LABEL parameter with the IMPORT or EXPORT batch command, the LABEL parameter in the JCL will be overridden.

For compatibility with the IBM Tape Management System, the label number ".Lnnn" is appended to the tape DSNAME as the last qualifier. If the combined length of the specified DSNAME and the appended label number exceeds the maximum length for DSNAME, an error message is issued.

MGMTCLAS

Specifies the eight-character management class name for the Storage Management Subsystem (SMS) operating environment. This parameter is only valid for new shared directories or transaction databases, or when the REALLOCATE parameter is set to YES.

MODEL

Use this parameter to specify the DSN of an existing shared directory or transaction database that can be used as a "template" for creating a new shared directory or transaction database with the same VSAM characteristics.

When the MODEL parameter is used, you can override any of the VSAM or BSAM parameters by including the appropriate parameter with the CREATE or EXPAND command. For example, to use a particular shared directory as a template for creating a new shared directory, but with a different CISIZE, include the CISIZE parameter in the CREATE command.

MONTHSOLD

Allows you to select entries in a shared directory or transaction database that are older than nnn months. For example, using the DELETE command for a transaction database with MONTHSOLD=6 deletes all transaction report entries older than six months.

NOTFOUND

Specifies whether to request a return code 00, instead of the default code 44, when an entry is not found. The valid values are:

ERROR: (Default) Allows the default code.

IGNORE: Forces a return code 00.

OBJECT (OBJ)

Specifies the type of object being changed, deleted, locked, or unlocked; or the type of object being created, reallocated, or formatted. This parameter also specifies the shared directory for the EXPAND command, and the shared directory or the transaction database for the DIRX command. Allowable values are:

SHRDIR: Shared directory

DATABASE: transaction database

An additional option available for the DELETE command is:

ENTRY: Member in the file. This option is only used with the DELETE command when removing an entry from a transaction database and the shared directory to which it is attached.

REALLOCATE (REALLO)

If the shared directory or transaction database you have specified already exists, you can use this parameter to reallocate the file. Allowable values are:

NO: (Default) For sequential files, the file will be reformatted if possible. For VSAM files, an error message is returned and the file is left unchanged.

YES: For both VSAM and sequential files, the file will be deleted and then reallocated.

RECORDS (RECS)

If you have specified AM=VSAM and want to use records as the unit of measure for the size of the file, use this parameter to specify the number of records. This parameter is valid for VSAM files only.

REPAIR

Attempts repair of any shared directory file structure errors and directory entry mismatches between the shared directory and all of its attached report databases.

REPTCOUNT (RC)

Specifies the maximum number of entries in the transaction database, provided there is enough space. The minimum value is 1, and the maximum number of Abend-AID for CICS transaction diagnostic reports that can exist in a transaction database is 13680. The default is 100.

REPTNUMBER (RN)

Specifies an entry's number. If this parameter is specified, all other selective parameters are ignored. Maximum length is ten digits.

RETAINRN

Specifies whether the entry's report number will be retained during a move or copy. Allowable values are:

NO (Default) Do not retain the number.

YES: Retain the number. In this case, if the number to be retained is already being used in the "to" file, then the next highest available number will be used. This value should only be used when moving or copying to a shared directory.

SECONDARY

This parameter can only be used when formatting pre-allocated shared directories or transaction databases. It allows shared directories or VSAM transaction databases to be allocated with secondary extents. Allocating secondary extents allows a shared directory or VSAM transaction database to be allocated on multiple volumes. The default is NO.

Note: This parameter is only valid for shared directories or VSAM transaction databases, and the VSAM cluster must be pre-allocated.

SORT

Specifies the order in which entries are selected for the execution of a command. By default, entries are selected in descending date/time order:.

- Entries are first ordered by date, from newest (most recent) to oldest.
- Entries with the same date are then ordered by time (from latest to earliest).

Valid sort keys, their abbreviations, and the sorting sequence they cause are specified in Table 5-2.

Table 5-2. Transaction Database Sort keys for Abend-AID for CICS

Sort Key	Abbreviation	Primary Sort Field	Sequence	Secondary Sort Field	Sequence	Tertiary Sort Field	Sequence
TRAN		TRAN	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
TERM		TERM	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
SYSTEM		SYSTEM	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
NETNAME	NET	NETNAME	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
NUMBER	NUM	REPT-NUMBER	Descending				
DATE		DATE/TIME	Descending	REPT-NUMBER	Descending		

STARTDATE (SDATE)

Specifies the processing date of the transaction reports. The STARTDATE parameter is used with the ENDDATE parameter to specify a date range.

The STARTDATE parameter can be specified as either MM/DD/YY or MM/DD/YYYY. Acceptable four-digit years are 1970 through 2069. When this parameter is used alone, all entries for the specified date are selected.

The earliest valid start date is 01/01/1970.

STARTTIME (STIME)

Specifies the beginning processing time of the transaction reports. This parameter must be specified as HH:MM[:SS] or HH.MM[.SS]. If seconds are omitted, the default is 00. This parameter must precede the ENDTIME parameter (if ENDTIME is specified).

If the STARTTIME parameter is specified and the ENDTIME parameter is omitted, then ENDTIME defaults to 24:00:00.

When using STARTTIME and ENDTIME in conjunction with STARTDATE and ENDDATE, entries are first selected by date, then by time.

STORCLAS

Specifies the eight-character storage class name for the Storage Management Subsystem (SMS) operating environment. This parameter is only valid for new shared directories or transaction databases, or when the REALLOCATE parameter is set to YES.

TODD

Specifies the ddname of the transaction database or shared directory to which the entries are copied, exported, or moved. It also specifies the ddname of the sequential file to which entries are exported.

TODSN

Specifies the DSN of the transaction database or shared directory to which the entries are copied, exported, or moved. It also specifies the DSN of the sequential file to which entries are exported.

TOOBJECT

Specifies the type of file to which the entries are copied, imported, or moved. Allowable values are:

SHRDIR: The “to” file is a shared directory.

DATABASE: The “to” file is a transaction database.

TOVOL

Specifies the VOLSER of the transaction database or shared directory to which the entries are copied, exported, imported, or moved. It also specifies the VOLSER of the sequential file to which previously exported entries are exported.

TRACKS (TRKS)

To use tracks as the unit of measure for the size of the file, use this parameter to specify the number of tracks. This parameter is valid for both VSAM and sequential files.

TRAN (T)

Specifies Abend-AID for CICS transaction reports by transaction ID or by transaction and terminal ID concatenation. Wildcard characters are valid.

```
TRAN=ABCDTT01 or TRAN=ABCDTT*
```

TYPE

If you are creating a transaction database, use this parameter to specify the type of entries the file can contain. For a Abend-AID for CICS transaction database, code TYPE=CICS.

UNIT

If you have specified AM=BSAM, you can use this parameter to specify a device or device type. The default is SYSDA.

VOLSER (VOL)

If you have specified OBJECT=DATABASE, you can use this parameter to specify the volume serial number of the transaction database when you are using the DSN parameter.

VSAMCAT

If you have specified AM=VSAM, you can use this parameter to specify the VSAM catalog in which the file will be defined. This is an optional parameter.

YEARSOLD

Allows you to select entries in a shared directory or transaction database that are older than nnn years. For example, using the DELETE command for a transaction database with YEARSOLD=1 deletes all transaction report entries that are older than 1 year.

Commands Descriptions and Syntax

This section describes the commands used to manage shared directories and transaction databases including valid command abbreviations and the parameters allowed for each command. With the exception of specifying REPTNUMBER, you can enter more than one parameter with a command.

ATTACH (ATT)

The ATTACH command connects a transaction database to a shared directory and populates the shared directory with transaction database entries. A transaction database must be attached to a shared directory before it can be used, and Abend-AID for CICS must be turned OFF in the CICS region during the attach.

The following diagram illustrates the syntax for the ATTACH command.

```
►►─ATTach──┬─DIRDD=ddname─┬─,─┬─DDName=ddname─┬──►◄
           └─DIRDSN=dsn───┘   └─DSN=dsn────────┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Attach a transaction database to a shared directory.

```
ATTACH DSN=COMPWARE.AACICS.SERVER1.TRANDB01,
       DIRDSN=COMPWARE.AACICS.SERVER1.SHRDIR
```

Note: If an error occurs during the ATTACH process, when adding members into the shared directory, the database will set “attach in progress” (will show on a DIRX), and the shared directory MAY contain entries for only some (but not all) of the database members. If the problem cannot be solved by another attempt to ATTACH, then perform the following to eliminate possible integrity problems:

Run DETACH on the database to clear the ATTACH setting and remove entries added during the ATTACH process.

CHANGE (CHG)

The CHANGE command changes the AUTODELETE attribute of a transaction database. For the change in the CICS region to take effect, Abend-AID for CICS must be turned OFF in all CICS regions before the change is made. It must be turned ON again in all CICS regions after the change is made to reopen the file and load the changed attribute.

The following diagram illustrates the syntax for the CHANGE command.

```
►►─CHanGe──OBJect=DATABASE, ─┬DDName=ddname──────────────┬, ──►
                             └DSN=dsn, ─┬──────────────┬─┘
                                        └VOLser=volser─┘
  ┌AutoDelete=YES─┐
►─┼───────────────┼──►◄
  ├AutoDelete=DUPS┤
  └AutoDelete=NO──┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Change the AUTODELETE attribute on a transaction database from AUTODELETE=NO to AUTODELETE=DUPS.

```
CHANGE OBJECT=DATABASE,DSN=COMPWARE.AACICS.SERVER1.TRANDB01,AD=DUPS
```

COPY

The COPY command copies selected entries in any of the following ways:

- from one transaction database to another transaction database
- from a transaction database to a shared directory
- from a shared directory to a transaction database
- from one shared directory to another shared directory.

Use the FROMDD and TODD parameters to specify the “from” and “to” file ddnames. When an entry is copied to or from a transaction database that is attached to a shared directory, the shared directory is also updated immediately. If no other parameters are specified, then all entries are copied.

The following diagram illustrates the syntax for the COPY command.

```
►►─COPY──┬─TOOBJECT=SHRDIR──────┬─,─┬─TODD=ABNLTO──────────────────────┬─,──►
         └─TOOBJECT=DATABASE────┘   ├─TODD=ddname──────────────────────┤
                                    └─TODSN=dsn,──┬─────────────────────┬┘
                                                  └─TOVOL=volser────────┘
►─┬─FROMOBJECT=SHRDIR──────┬─,─┬─FROMDD=ABNLFROM──────────────────────┬─,──►
  └─FROMOBJECT=DATABASE────┘   ├─FROMDD=ddname──────────────────────┤
                               └─FROMDSN=dsn,──┬─────────────────────┬┘
                                               └─FROMVOL=volser──────┘
►─┬─RETAINRN=NO──────┬─,──────────────────────────────────────────────────
  └─RETAINRN=YES─────┘

──┬──────────────────────────────┬─,──┬──────────────────────┬─,──┬──────────────┬─,──►
  └─ReptNumber=xxxxxxxxxxx───────┘    └─Tran=xxx*────────────┘    └─APPLID=xxxxxxx*─┘

──┬──────────────────────────────┬─,──┬──────────────────────┬─,──►
  └─CicsCode=xxx*────────────────┘    └─SORT=sortkey─────────┘

►─┬─DAYSOLD=nnn──────────────────────────────────────────────────────────┬─,──►
  ├─MONTHSOLD=nnn─────────────────────────────────────────────────────────┤
  ├─YEARSOLD=nnn──────────────────────────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,──┬──────────────────────────────────────────────┬┘
                           └─EndDATE=mm/dd/yyyy───────────────────────────┘

►─┬──────────────────────────────────────────────────────────────────────┬─,──►
  └─StartTIME=hh:mm[:ss],──┬─EndTIME=24:00:00──────────────────────────┬┘
                           └─EndTIME=hh:mm[:ss]────────────────────────┘

►─┬─DUPS=ALL──────────┬─,──┬─INC=YES──────┬─,──┬─CONTINUE=YES──────┬─,──┬─NOTFOUND=ERROR──────┬──►◄
  ├─DUPS=LOCKED───────┤    ├─INC=NO───────┤    └─CONTINUE=NO───────┘    └─NOTFOUND=IGNORE───────┘
  └─DUPS=UNLOCKED─────┘    └─INC=ONLY─────┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Copy all entries from a transaction database to another transaction database.

```
COPY TOOBJECT=DATABASE,TODSN=COMPWARE.AACICS.SERVER1.TRANDB01,
     FROMOBJECT=DATABASE,
     FROMDSN=COMPWARE.AACICS.SERVER2.TRANDB02
```

CREATE (CRE)

The CREATE command performs the following functions:

- Allocates and formats a new shared directory
- Formats a pre-allocated shared directory
- Allocates and formats a new transaction database.
- Formats a pre-allocated transaction database.

CAUTION:
This command erases all of the information in the specified file.

Only VSAM files can be used for shared directory files; however, either VSAM or sequential files can be used for transaction databases.

When you create a transaction database, it is automatically connected to a shared directory that is specified as a parameter of the CREATE command. A transaction database must be connected to a shared directory before it can be used.

Sample JCL Members

Sample JCL members for the CREATE command are available in CSS installation library members. Please refer to Table 5-3 for a list of the sample members.

Table 5-3. Sample JCL Members for CREATE Command

To format...	Use sample member...
Sequential Transaction Databases	CXALLBSD
VSAM Transaction Databases	CXALLVSD
Shared Directories	CXALLMC

The following diagrams illustrate the syntax for the CREATE command based on the function you want to perform.

Allocating and Formatting a New Shared Directory:

The first CREATE syntax diagram illustrates the syntax for allocating and formatting a new shared directory.

```
►►─CREate─OBJect=SHRDIR,──┬─DIRDD=ddname──────────────────┬─,──►
                          └─DIRDSN=dsn,─┬──────────────┬─┘
                                        └─DIRVOL=volser─┘

►─AM=VSAM,──┬──────────────┬─,──┬──────────────────┬─,──►
            ├─RECords=nnnn─┤    │ ┌─CISIZE=4096─┐   │
            ├─TRacKS=nnnn──┤    ├─┼─────────────┼───┤
            ├─CYLinderS=nnnn┤    │ └─CISIZE=nnnn─┘   │
            └─DIRENTS=nnnn─┘    │ ┌─BLKsize=4089─┐   │
                                └─┼──────────────┼───┘
                                  └─BLKsize=nnnn─┘

                                    ┌─REALLOcate=NO─┐
►─┬──────────────┬─,─┬──────────┬─,─┼───────────────┼─,──►
  └─VSAMCAT=ddname─┘ └─MODEL=dsn─┘ └─REALLOcate=YES─┘

►─┬──────────────┬─,─┬──────────────┬─,─┬──────────────┬──►◄
  └─MGMTCLAS=xxxxxxxx─┘ └─STORCLAS=xxxxxxxx─┘ └─DATACLAS=xxxxxxxx─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Create a shared directory with a CISIZE of 24569, which can occupy five cylinders.

```
CREATE OBJECT=SHRDIR,DIRDSN=COMPWARE.AACICS.SERVER1.SHRDIR,
       AM=VSAM,CYLINDERS=5,CISIZE=24569,DIRVOL=ABCDEF
```

Formatting a Pre-Allocated Shared Directory

This CREATE syntax diagram illustrates the syntax for a pre-allocated shared directory.

```
                                                  ┌SECONDARY=NO──┐
►►─CREate─OBJect=SHRDIR,─┬DIRDD=ddname──┐ ,─┼──────────────┼──►◄
                         └DIRDSN=dsn────┘   └SECONDARY=YES──┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Format a pre-allocated shared directory and specify use of secondary extents.

```
CREATE OBJECT=SHRDIR,
       DIRDSN=COMPWARE.AACICS.SERVER1,SHRDIR,
       SECONDARY=YES
```

Allocating and Formatting a New Transaction Database

This CREATE syntax diagram illustrates the syntax for allocating and formatting a new transaction database.

```
►►─CREate──OBJect=DATABASE,──┬─DIRDD=ddname─┬─(1),─┬─DDName=ddname──────────┬─,──►
                             └─DIRDSN=dsn───┘      └─DSN=dsn─,─┬──────────────┬─┘
                                                                └─VOLser=volser─┘

►─┬─AM=VSAM,──┬──────────────┬─,──┬──────────────┬─,──┬──────────────┬─┬─,──►
  │           ├─RECords=nnnn─┤    │ ┌─CISIZE=4096─┐ │  └─VSAMCAT=ddname─┘ │
  │           ├─TRacKS=nnnn─┤    ├─┼─────────────┼─┤                     │
  │           └─CYLinderS=nnnn┘   │ └─CISIZE=nnnn─┘ │                     │
  │                               │ ┌─BLKsize=4089─┐ │                     │
  │                               ├─┼──────────────┼─┘                     │
  │                               └─BLKsize=nnnn─┘                        │
  │                                                                       │
  └─AM=BSAM,──┬──────────────┬─,──┬─UNIT=SYSDA─────┬─,──┬─BLKsize=4096─┬─┘
              ├─TRacKS=nnnn─┤    └─UNIT=xxxxxxxx─┘    └─BLKsize=nnnn─┘
              └─CYLinderS=nnnn┘

►─┬──────────┬─,──┬─REALLOcate=NO─┬─,──┬──────────────┬─,──►
  └─MODEL=dsn─┘    └─REALLOcate=YES─┘    └─MGMTCLAS=xxxxxxxx─┘

►─┬──────────────┬─,──┬──────────────┬─,──┬─ReptCount=100─┬─,──►
  └─STORCLAS=xxxxxxxx─┘    └─DATACLAS=xxxxxxxx─┘    └─ReptCount=nnnn─┘

►─TYPE=CICS,──┬─GroupCount=4──┬─,──┬─AutoDelete=YES──┬─,──┬─EXTENTS=64──┬──►◄
              └─GroupCount=nn─┘    ├─AutoDelete=DUPS─┤    └─EXTENTS=nnn─┘
                                   └─AutoDelete=NO───┘
```

Notes:

1. Please note that the DIRDD parameter (or DIRDSN) specifies the name of the shared directory to which you are attaching the transaction database. The DDNAME parameter (DSN) specifies the name of the transaction database.

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Create a sequential transaction database with the following characteristics:

- The maximum number of entries in the file is 50.
- At least one entry of each name will remain in the transaction database.
- When additional space is required in the transaction database, one block will be allocated.
- The maximum number of extents to be allocated to each transaction database entry is 256.
- The transaction database can occupy 5 cylinders.
- The blocksize is 24569.

```
CREATE OBJECT=DATABASE,DIRDSN=COMPWARE.AACICS.SERVER1.SHRDIR,
DSN=COMPWARE.AACICS.SERVER1.TRANDB01,VOLSER=PRD924,TYPE=CICS,
RC=50,AD=DUPS,GC=1,EXTENTS=256,AM=BSAM,CYLINDERS=5,BLKSIZE=24569
```

Formatting a Pre-Allocated Transaction Database

This CREATE syntax diagram illustrates the syntax for formatting a pre-allocated transaction database.

```
▶▶─CREate──OBJect=DATABASE,─┬─DIRDD=ddname─┬─,─┬─DDName=ddname─┬─,──▶
                            └─DIRDSN=dsn───┘   └─DSN=dsn───────┘

▶─┬─SECONDARY=NO──┬─,─┬─ReptCount=100──┬─,─TYPE=CICS,─┬─GroupCount=4──┬─,──▶
  └─SECONDARY=YES─┘   └─ReptCount=nnn──┘              └─GroupCount=nn─┘

▶─┬─AutoDelete=YES──┬─,─┬─EXTENTS=64──┬──────────────▶◀
  ├─AutoDelete=DUPS─┤   └─EXTENTS=nnn─┘
  └─AutoDelete=NO───┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Format a pre-allocated transaction database with the following characteristics:

- The maximum number of entries in the file is 100.
- Autodelete entries when full.
- When additional space is required in the transaction database, two blocks will be allocated.
- The maximum number of extents to be allocated to each transaction database entry is 64.

```
CREATE OBJECT=DATABASE,
       DIRDSN=COMPWARE.AACICS.SERVER1.SHRDIR,
       DSN=COMPWARE.AACICS.SERVER1.TRANDB01,TYPE=CICS,
       RC=100,GC=2,AD=YES,EXTENTS=64
```

DELETE (DEL)

The DELETE command deletes selected transaction database entries, a selected transaction database, or a selected shared directory.

Note: The DELETE command, while it removes records, may or may not free up any of the allocated blocks in use depending on whether other data are sharing the same blocks as the records being deleted.

DELETE an ENTRY from Both the Shared Directory and Transaction Database

DELETE removes the entry from both the shared directory and the transaction database. Transaction database entries can only be deleted if attached to a shared directory. You must specify either the DIRDD or the DIRDSN parameter. (To delete an entry, specify OBJECT=ENTRY.)

```
►►─DELete──OBJect=ENTRY,─┬─DIRDD=ddname─┬─,─┬──────────────────────┬─,──►
                         └─DIRDSN=dsn───┘   └─ReptNumber=xxxxxxxxxx─┘

►─┬──────────┬─,─┬──────────────┬─,─┬──────────────────┬─,──►
  └─Tran=xxx*─┘   └─CicsCode=xxx*─┘   └─APPLID=xxxxxxx*─┘

►─┬─DAYSOLD=nnn──────────────────────────────────────┬─,──►
  ├─MONTHSOLD=nnn────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,─┬──────────────────────────┘
                          └─EndDATE=mm/dd/yyyy─┘

►─┬──────────────────────────────────────────────────┬─,─┬─DUPS=ALL────────┬─,──►
  │                                                  │   ├─DUPS=LOCKED────┤
  └─StartTIME=hh:mm[:ss],─┬─EndTIME=24:00:00─┬─────┘   └─DUPS=UNLOCKED──┘
                          └─EndTIME=hh:mm[:ss]─┘

►─┬─INC=YES──┬─,─┬─NOTFOUND=ERROR──┬──►◄
  ├─INC=NO───┤   └─NOTFOUND=IGNORE─┘
  └─INC=ONLY─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Delete all entries with an APPLID of CICSAPPL.

```
DELETE OBJECT=ENTRY,
       DIRDSN=COMPWARE.AACICS.SERVER1.SHRDIR,
       APPLID=CICSAPPL
```

DELETE a Transaction Database (and Detach It from the Shared Directory)

If the transaction database is attached to a shared directory, DELETE also detaches it from the shared directory prior to deletion. You must specify either the DDNAME or the DSN parameter. (To delete a transaction database, specify OBJECT=DATABASE.)

```
►►─DELete──OBJect=DATABASE,─┬─DDName=ddname──────────────┬──►◄
                            └─DSN=dsn,─┬──────────────┬─┘
                                       └─VOLser=volser─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Delete a transaction database (and detach it from the shared directory).

```
DELETE OBJECT=DATABASE,
       DSN=COMPWARE.AACICS.SERVER1.TRANDB01
```

DELETE a Shared Directory

When deleting a shared directory, you can choose whether to also delete all attached transaction databases. Regardless of your choice, a DETACH is performed on all attached transaction databases prior to the deletion. You must specify either the DIRDD or the DIRDSN parameter. (To delete a shared directory, specify OBJECT=SHRDIR.)

```
                                          ┌─DataBaseDISP=DETACH─┐
►►─DELete─OBJect=SHRDIR,─┬─DIRDD=ddname─┬─,─┤                     ├──►◄
                         └─DIRDSN=dsn───┘   └─DataBaseDISP=DELETE─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Examples

Delete a shared directory and also delete any transaction databases attached to it.

```
DELETE OBJECT=SHRDIR,
       DSN=COMPWARE.AACICS.SERVER1.SHRDIR,DBDISP=DELETE
```

Delete a shared directory, but first detach any transaction databases attached to it (does NOT delete databases).

```
DELETE OBJECT=SHRDIR,
       DSN=COMPWARE.AACICS.SERVER1.SHRDIR,DBDISP=DETACH
```

DETACH

The DETACH command disconnects a transaction database from a shared directory and removes the transaction database entries from the shared directory. Prior to issuing the DETACH command, the transaction database must be UNASSIGNED in the Abend-AID for CICS viewer. Then Abend-AID for CICS must be turned OFF in all CICS regions and the server brought down.

To ensure that you detach the transaction database from the correct shared directory, you may optionally use the DIRDSN parameter to specify the name of the shared directory. If the shared directory is specified incorrectly, an error will be returned. If the DIRDSN is not specified, then the transaction database will be unconditionally detached from the appropriate shared directory.

The following diagram illustrates the syntax for the DETACH command.

```
>>-DETACH---+-DDName=ddname-----------+-,-+-----------+-->
            +-DSN=dsn,-+-------------+-+   +-DIRDSN=dsn-+
                       +-VOLser=volser-+
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Detach a transaction database from a shared directory.

```
DETACH DDNAME=TRFILE1
```

Note: If an error occurs during the DETACH process, when deleting members from the shared directory, the database will set “detach in progress” (will show on a DIRX) but will still appear attached, and the shared directory may still contain entries for members on the database that had not yet been removed from the shared directory. To eliminate possible integrity problems perform the following:

1. Run RESET on the database to clear the attach setting.
2. Run SCRATCH on the shared directory for that database to remove any entries left for that database.

DIRECTORY (DIR, D) and DIRX (DX)

The DIRECTORY command lists selected entries in a shared directory or transaction database.

The DIRX command functions similarly to the DIRECTORY command, but it also displays attribute and allocation information. When DIRX is performed on a shared directory, it displays the attribute and allocation information for the shared directory and all transaction databases attached to that shared directory.

The following diagram illustrates the syntax for the DIRECTORY or DIRX command.

```
►►─DIRectory or DIRX─┬──────────────────────┬─,──►
                     └ReptNumber=xxxxxxxxxx─┘

►─┬─OBJect=SHRDIR,─┬─DIRDD=ddname─┬──────────────┬─,──►
  │                └─DIRDSN=dsn───┘              │
  └─OBJect=DATABASE,─┬─DDName-ddname──────────────┤
                     └─DSN=dsn,─┬──────────────┬─┘
                                └VOLser=volser─┘

►─┬──────────┬─,─┬──────────┬─,─┬──────────────┬─,──►
  └Tran=xxx*─┘   └CicsCode=xxx*┘   └APPLID=xxxxxxx*┘

►─┬──────────────────────────────────────────────┬─,──►
  ├DAYSOLD=nnn──────────────────────────────────┤
  ├MONTHSOLD=nnn────────────────────────────────┤
  ├YEARSOLD=nnn─────────────────────────────────┤
  └StartDATE=mm/dd/yyyy,─┬──────────────────┬──┘
                         └EndDATE=mm/dd/yyyy─┘

►─┬──────────────────────────────────────────────┬─,──►
  │                        ┌EndTIME=24:00:00───┐  │
  └StartTIME=hh:mm[:ss],─┼──────────────────┼──┘
                         └EndTIME=hh:mm[:ss]─┘

  ┌DUPS=ALL───────┐   ┌INC=YES───┐   ┌NOTFOUND=ERROR───┐
►─┼───────────────┼─,─┼──────────┼─,─┼─────────────────┼─,──►
  ├DUPS=LOCKED────┤   ├INC=NO────┤   └NOTFOUND=IGNORE──┘
  └DUPS=UNLOCKED──┘   └INC=ONLY───┘

►─┬──────────────┬──►◄
  └SORT=sortkey─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5. Refer to Shared Services SLXCNTL installation library member CXFXDIRX for sample JCL for the DIRX command.

Example 1

List all entries in a shared directory.

```
DIR OBJECT=SHRDIR,DIRDSN=COMPWARE.AACICS.SERVER1.SHRDIR
```

Example 2

List all entries in a transaction database sorted by transaction name (TRAN), and include attribute and allocation information.

```
DIRX OBJECT=DATABASE,DSN=COMPWARE.AACICS.SERVER1.TRANDB01,
     SORT=TRAN
```

DIRECTORY SEQUENTIAL (DIRS)

The DIRS (DIRectory Sequential) command lists selected directory entries in a sequential file produced by the EXPORT command.

If no other parameters are specified, then all entries are listed.

The following diagram illustrates the syntax for the DIRECTORY SEQUENTIAL command.

```
>>-DIRS---+-DDName=ddname-------------------+-,-------->
          +-DSN=dsn,-+-------------------+-+
                     +-VOLser=volser-----+
>-+-------------------------+-,-+-------------+-,-+-------------------+-,-->
  +-ReptNumber=xxxxxxxxxx-+   +-Tran=xxx*-+   +-CicsCode=xxx*-+
>-+-------------------------------------------------------------+-,-->
  +-DAYSOLD=nnn-------------------------------------------------+
  +-MONTHSOLD=nnn-----------------------------------------------+
  +-YEARSOLD=nnn------------------------------------------------+
  +-StartDATE=mm/dd/yyyy,-+-------------------------+-----------+
                          +-EndDATE=mm/dd/yyyy-+
>-+-------------------------------------------------------------+-,-->
  +-StartTIME=hh:mm[:ss],-+-EndTIME=24:00:00-----+-------------+
                          +-EndTIME=hh:mm[:ss]-+
>-+-DUPS=ALL-------+-,-+-INC=YES-----+-,-+-NOTFOUND=ERROR-----+-,-->
  +-DUPS=LOCKED----+   +-INC=NO------+   +-NOTFOUND=IGNORE----+
  +-DUPS=UNLOCKED--+   +-INC=ONLY----+
>-+-------------+-,-+-LABEL=1-----+-->>
  +-SORT=sortkey-+   +-LABEL=nn---+
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Export all unlocked entries and list all directory entries on the resulting sequential file.

```
EXPORT DUPS=UNLOCKED
DIRS DSN=EFHBRR0.EXFILE1
```

ESTIMATE

The ESTIMATE command calculates a reasonable number of blocks to allocate for a shared directory when the allocation is performed outside of CSS (i.e., through IDCAMS or Compuware's File-AID product). The resulting calculation is expressed in records.

The following diagram illustrates the syntax for the ESTIMATE command.

```
►►─ESTIMATE─OBJect=SHRDIR, ─┬─BLKsize=nnnn─┬─,─┬──────────────┬─►◄
                            ├─CISIZE=nnnn──┤   └─DIRENTS=nnnn─┘
                            └─MODEL=dsn────┘
```

For a description of the parameters see "Command Parameters for Shared Directories and Transaction Databases" on page 5-5.

Example

Determine the block allocation for a shared directory with a maximum of 200 entries and a CISIZE of 24569.

```
ESTIMATE OBJECT=SHRDIR,CISIZE=24569,DIRENTS=200
```

EXAMINE

The EXAMINE command reads, analyzes, and **optionally** repairs the selected shared directory to ensure the following:

- Internal shared directory structure integrity. Shared directory index and data components match. That is, all entries in the shared directory are accessible via an indexed search, and are selectable for abend diagnosis viewing.
- All transaction databases attached to the shared directory still exist in the MVS catalog.
- All entries in the shared directory exist in the attached transaction databases and can be opened for abend diagnosis.

Note: This command creates a report of any inconsistencies and **optionally with the REPAIR directive specified**:

- repairs shared directory index pointers,
- deletes, from the shared directory, orphaned data entries that are not selectable via the indexed path,
- deletes members from the shared directory that no longer exist in the attached database files
- deletes members from the attached databases, that are no longer in the shared directory
- deletes incomplete members in the shared directory and attached databases (if DELETEIC is specified).

The following diagram illustrates the syntax for the EXAMINE command.

```
►►--EXAMINE──┬─DIRDD=ddname──┬─┬──────────┬─┬──────────┬──►◄
             └─DIRDSN=dsn────┘ └─REPAIR───┘ └─DELETEIC─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Note: The REPAIR directive attempts exclusive ENQ control of the selected shared directory. Any Abend-AID for CICS dump capture (TDCAS) and/or viewing servers **must** be stopped or examine/repair will not proceed.

EXAMINE may be run any time, **without** the REPAIR directive, to analyze the shared directory’s integrity. However, during periods of dump capture activity, INDEX/DATA mismatches may be erroneously reported on the EXAMINE report.

An effort was made to prevent EXAMINE/REPAIR from rendering the shared directory or its contents, unusable or impaired after REPAIR processing. However, a backup of the shared directory before using REPAIR is advised.

Runtime Example

```
EXAMINE DIRDSN=COMPUWARE.AACICS.SERVER1.SHRDIR
```

Figure 5-1. Sample EXAMINE Report

```
SHARED DIRECTORY EXAMINE REPORT FOR: CX.TECH.P120666.SDIR9                                                    PAGE:   1

                        DUMP   TRAN ABND               ABEND                        SMF
      ERROR            NUMBER  NAME CODE PROGRAM NAME  OFFSET  JOBNAME    APPLID   ID          TIMESTAMP
-------------------- --------- ---- ---- ------------ -------- ---------  ------- ------ ----------------------------------
n/a via Index search       228 R007 0100 RSY0005       00181AA  CICSEEPF   CICSPF  SYSA      8.15.10.556470  09-01-2000
...DATA RBN: 00000015 contains abend report above that cannot be accessed via the Shared Directory Index

n/a via Index search     1,081 X830 R1II RXC0003       0004D38  CICSEEPA   CICSPA  SYSA     14.50.31.338171  09-08-2000
...DATA RBN: 00000015 contains abend report above that cannot be accessed via the Shared Directory Index

Rpt. not on Database     1,943 A620 54GE RQUL620       000AAEE  CICSEEPB  CICSPB   SYSA     10.51.01.677688  09-15-2000
  SYS3.RCI.AACICS42.SERVERA.TRANDB02
Rpt. not on Database     1,991 B000 ATND               0000000  CICSEEPT  CICSPT   SYSA     14.53.58.747975  09-15-2000
  SYS3.RCI.AACICS42.SERVERA.TRANDB02
Rpt. not on Database     1,987 B200 ATNI               0000000  CICSEEPT  CICSPT   SYSA     14.42.50.242297  09-15-2000
  SYS3.RCI.AACICS42.SERVERA.TRANDB02


SHARDDIR DATA BLOCKS READ---------6  SHARDDIR DATA RECORDS PROCESSED--------60
SHARDDIR INDX BLOCKS READ---------3  SHARDDIR INDX RECORDS PROCESSED--------66


SHARED DIRECTORY PROCESSING ERROR  FOUND



*** ALL FUNCTIONS SUCCESSFULLY COMPLETED
```

EXPAND

The EXPAND command reallocates an existing shared directory using new allocation parameters and copies all of the entries to the newly allocated shared directory. After the copy has completed successfully, the old shared directory is renamed to a temporary name, the new shared directory is renamed to the original name, and the old shared directory is then deleted. You can use this command to increase or decrease the space allocated for a shared directory and/or change the CISIZE.

When using the EXPAND command, the following rules apply:

- The shared directory must already exist (i.e., you cannot use EXPAND to create a new shared directory).
- The shared directory must not be in use (i.e., you must stop the server and turn OFF Abend-AID for CICS in all CICS regions).

Use the DIRDD or DIRDSN parameter to specify the existing shared directory. You can optionally use the VOLSER to specify the desired volume location for the new expanded shared directory.

The following diagram illustrates the syntax for the EXPAND command.

```
►►─EXPAND─OBJect=SHRDIR,─┬─DIRDD=ddname─┬─,─┬──────────────┬─,──►
                         └─DIRDSN=dsn───┘   └─VOLser=volume─┘

                                                ┌─CISIZE=4096─┐
►─┬──────────────────┬─,─┬──────────────────┬─,─┼─────────────┼─,─┬──────────────┬─►◄
  └─VSAMCAT=ddname───┘   ├─RECords=nnnn───┤   └─CISIZE=nnnn───┘   └─MODEL=dsn───┘
                         ├─TRacKS=nnnn───┤
                         └─CYLinders=nnnn┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Expand an existing shared directory, but change the CISIZE to 8192.

```
EXPAND OBJECT=SHRDIR,DIRDD=CAT11,
       MODEL=COMPWARE.AACICS.SERVER1.SHRDIR,
       CISIZE=8192
```

EXPORT

The EXPORT command exports selected entries from a shared directory or transaction database to a sequential file, from which they can later be imported into the same or another shared directory or transaction database using the IMPORT command.

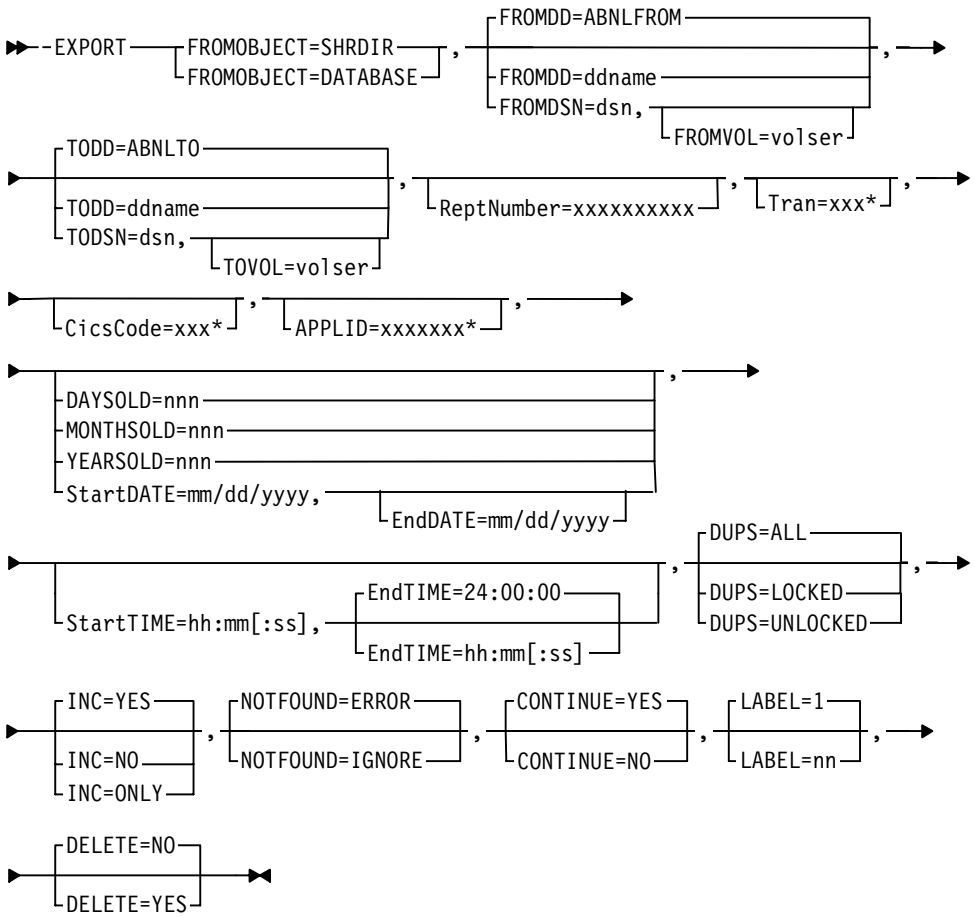
The “from” file (the shared directory or transaction database) can be specified using either the FROMDSN or FROMDD parameter.

The “to” file (the sequential file) can be specified using either the TODSN and TOVOL parameters, or by using the TODD parameter. However, if you are exporting to a tape, you **must** use the TODD parameter and not the TODSN and TOVOL parameters. (The ABNLFROM and ABNLTO DD statements may be used for exporting.) When an entry is exported from a transaction database and DELETE=YES, the shared directory is also updated.

If no other parameters are specified, then all entries are exported.

You may use multiple EXPORT commands if you include a unique TODD (or TODSN and TOVOL) parameter for each output dataset or PDS member that is a target of the export. If unique parameters are not used, each successive export will overlay the previous export.

The following diagram illustrates the syntax for the EXPORT command.



For a description of the parameters see "Command Parameters for Shared Directories and Transaction Databases" on page 5-5. Refer to Shared Services SLXCNTL installation library member CXFXEXPO for sample JCL for the EXPORT command.

Example

For a transaction database, export all complete transaction reports for the transaction IDs starting with ABCD and abended with CICS transaction abend code ASRA to a tape. Delete all entries that have been successfully exported.

```
EXPORT FROMOBJECT=DATABASE,FROMDD=ABNLFROM,TODD=ABNLTO,
       TRAN=ABCD*,CICSCODE=ASRA,INC=NO,DELETE=YES
```

IMPORT

The IMPORT command selectively restores previously exported entries from a sequential file to a shared directory or transaction database.

The “from” file (the sequential file) can be specified using either the FROMDSN and FROMVOL parameters, or by using the FROMDD parameter.

The “to” file (the shared directory or transaction database) can be specified using either the TODSN and TOVOL parameters, or by using the TODD parameter.

When an entry is imported to a transaction database that is attached to a shared directory, the shared directory is also updated.

Note: If the access method is **VSAM**, then you must specify **DISP=SHR** for the TODSN DD statement in your JCL.

If no other parameters are specified, then all entries are imported.

The following diagram illustrates the syntax for the IMPORT command.

```
>>--IMPORT---+-TOOBJECT=SHRDIR----+,--->
             +-TOOBJECT=DATABASE---+

 +-FROMDD=ABNLFROM----------------+,--+-TODD=ABNLTO----------------+,--->
 |                                |   |                            |
 +-FROMDD=ddname ---------------+   +-TODD=ddname----------------+
 +-FROMDSN=dsn,-+-----------+-+     +-TODSN=dsn,-+-----------+-+
                +-FROMVOL=volser-+                +-TOVOL=volser-+

>--+-----------------+,--+-------+,--+-----------+,--->
   +-ReptNumber=xxxxxxxxxx-+   +-Tran=xxx*-+   +-CicsCode=xxx*-+

>--+-------------------------------------------+,--->
   +-DAYSOLD=nnn-----------------------------+
   +-MONTHSOLD=nnn---------------------------+
   +-YEARSOLD=nnn----------------------------+
   +-StartDATE=mm/dd/yyyy,-+-----------+-+
                           +-EndDATE=mm/dd/yyyy-+

>--+-------------------------------------------+,--+-DUPS=ALL-------+,--->
   +-StartTIME=hh:mm[:ss],-+-EndTIME=24:00:00-+-+   +-DUPS=LOCKED----+
                           +-EndTIME=hh:mm[:ss]-+   +-DUPS=UNLOCKED--+

>--+-INC=YES-----+,--+-NOTFOUND=ERROR-+,--+-CONTINUE=YES-+,--+-LABEL=1-+--><
   +-INC=NO------+   +-NOTFOUND=IGNORE-+   +-CONTINUE=NO--+   +-LABEL=nn-+
   +-INC=ONLY----+
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5. Refer to Shared Services SLXCNTL installation library member CXFXIMPO for sample JCL for the IMPORT command.

Example

Restore all entries from a sequential file to a transaction database.

```
IMPORT TOOBJECT=DATABASE,FROMDSN=EFHBMM0.DB01,
       TODSN=COMPWARE.AACICS.SERVER1.TRANDB01
```

LOCK

The LOCK command locks selected entries of a specified transaction database or all transaction databases attached to a specified shared directory.

While an entry is locked with this command (i.e., manually locked), it cannot be automatically deleted when the transaction database becomes full.

The following diagram illustrates the syntax for the LOCK command.

```
►►─LOCK──┬─OBJect=SHRDIR, ──┬─DIRDD=ddname──┬──────────────┬──────────────┬──►
         │                  └─DIRDSN=dsn────┘              │
         └─OBJect=DATABASE,──┬─DDName=ddname──────────────┬──┘
                             └─DSN=dsn,──┬──────────────┬─┘
                                         └─VOLser=volser─┘

►─┬──────────────────────┬─┬──────────────┬─┬──────────────┬──►
  └─ReptNumber=xxxxxxxxxx─┘ └─Tran=xxx*────┘ └──────────────┘

►─┬──────────────┬─┬──────────────────┬─┬──────────────┬──►
  └─CicsCode=xxx*─┘ └─APPLID=xxxxxxx*──┘ └──────────────┘

►─┬──────────────────────────────────────────────────────┬─┬──────────────┬──►
  ├─DAYSOLD=nnn──────────────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,──┬──────────────────────────┬──┘
                           └─EndDATE=mm/dd/yyyy───────┘

►─┬──────────────────────────────────────────────────────┬─┬──────────────┬──►
  │                          ┌─EndTIME=24:00:00──────┐   │
  └─StartTIME=hh:mm[:ss],──┬─┴──────────────────────┴──┘
                           └─EndTIME=hh:mm[:ss]────────┘

  ┌─DUPS=ALL──────┐  ┌─INC=YES──┐  ┌─NOTFOUND=ERROR──┐
►─┼───────────────┼─┼──────────┼─┼─────────────────┼──►◄
  ├─DUPS=LOCKED───┤  ├─INC=NO───┤  └─NOTFOUND=IGNORE──┘
  └─DUPS=UNLOCKED─┘  └─INC=ONLY─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

For a transaction database, lock transaction report number 1105.

```
LOCK OBJECT=DATABASE,DSN=COMPWARE.AACICS.SERVER1.TRANDB01,
     REPTNUMBER=1105
```

MOVE

The MOVE command copies selected entries from one shared directory or transaction database to another shared directory or transaction database and deletes them from the "from" file if they were successfully copied. Entries can be moved in any of the following ways:

- from one transaction database to another transaction database
- from a transaction database to a shared directory
- from a shared directory to a transaction database
- from one shared directory to another shared directory.

Use the FROMDD and TODD parameters to specify the "from" and "to" file ddnames.

When an entry is moved to or from a transaction database that is attached to a shared directory, the shared directory is also updated.

If no other parameters are specified, then all entries are moved.

Note: Only successfully copied entries are deleted from the "from" file.

The following diagram illustrates the syntax for the MOVE command.

```
                                   ┌TODD=ABNLTO──────────────┐
►►─MOVE─┬TOOBJECT=SHRDIR──┬,─┼──────────────────────────┼,──►
        └TOOBJECT=DATABASE┘   ├TODD=ddname──────────────┤
                              └TODSN=dsn,─┬──────────────┘
                                          └TOVOL=volser┘
                                 ┌FROMDD=ABNLFROM──────────────┐
►──┬FROMOBJECT=SHRDIR──┬,─┼──────────────────────────────┼,──►
   └FROMOBJECT=DATABASE┘   ├FROMDD=ddname──────────────┤
                           └FROMDSN=dsn,─┬──────────────┘
                                         └FROMVOL=volser┘
   ┌RETAINRN=NO─┐
►──┼────────────┼,─┬──────────────────────┬,─┬──────────┬,──►
   └RETAINRN=YES┘   └ReptNumber=xxxxxxxxxx┘   └Tran=xxx*┘
►──┬──────────────┬,─┬──────────────┬,─┬──────────────┬,──►
   └APPLID=xxxxxxx*┘   └CicsCode=xxx*┘   └SORT=sortkey┘
►──┬──────────────────────────────────────────────────┬,──►
   ├DAYSOLD=nnn──────────────────────────────────────┤
   ├MONTHSOLD=nnn────────────────────────────────────┤
   ├YEARSOLD=nnn─────────────────────────────────────┤
   └StartDATE=mm/dd/yyyy,─┬──────────────────────┬──┘
                          └EndDATE=mm/dd/yyyy────┘
►──┬──────────────────────────────────────────────────┬,──►
   │                          ┌EndTIME=24:00:00──┐   │
   └StartTIME=hh:mm[:ss],─┼──────────────────────┼──┘
                            └EndTIME=hh:mm[:ss]────┘
   ┌DUPS=ALL──────┐   ┌INC=YES──┐   ┌CONTINUE=YES─┐   ┌NOTFOUND=ERROR──┐
►──┼──────────────┼,─┼─────────┼,─┼─────────────┼,─┼────────────────┼──►◄
   ├DUPS=LOCKED───┤   ├INC=NO───┤   └CONTINUE=NO──┘   └NOTFOUND=IGNORE──┘
   └DUPS=UNLOCKED─┘   └INC=ONLY─┘
```

For a description of the parameters see "Command Parameters for Shared Directories and Transaction Databases" on page 5-5. Refer to Shared Services SLXCNTL installation library member CXFXMOVE for sample JCL for the MOVE command.

Examples

Move all entries from one shared directory to another, and do not retain the report numbers.

```
MOVE TOOBJECT=SHRDIR,TODSN=COMPWARE.AACICS.SERVER2.SHRDIR,
     FROMOBJECT=SHDIR,FROMDSN=COMPWARE.AACICS.SERVER1.SHRDIR,
     RETAINRN=NO
```

Move all unlocked entries from one transaction database to another transaction database, for example, if you are moving the reports from the database to a file attached to a new server.

```
MOVE TOOBJECT=DATABASE,TODSN=COMPWARE.AACICS.SERVER2.TRANDB02,
     FROMOBJECT=DATABASE,
     FROMDSN=COMPWARE.AACICS.SERVER1.TRANDB01,
     DUPS=UNLOCKED
```

RESET

The RESET command disconnects a transaction database from a shared directory when the shared directory is lost or damaged. The transaction database can then be attached to another shared directory.

The following diagram illustrates the syntax for the RESET command.

```
►►─RESET──┬─DDName=ddname──┬──►◄
          └─DSN=dsn────────┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Disconnect a transaction database from a damaged shared directory by resetting all the flags.

```
RESET DSN=COMPWARE.AACICS.SERVER1.TRANDB01
```

SCRATCH (SCR)

The SCRATCH command clears all entries from a shared directory for a specified transaction database that has been lost or damaged.

The following diagram illustrates the syntax for the SCRATCH command.

```
►►─SCRatch──┬─DIRDD=ddname  ┌─,-DSN=dsn──►◄
            └─DIRDSN=dsn────┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Scratch all entries from a shared directory for a lost transaction database.

```
SCRATCH DIRDSN=COMPWARE.AACICS.SERVER1.SHRDIR,
        DSN=COMPWARE.AACICS.SERVER1.TRANDB01
```

UNLOCK (UNLK)

The UNLOCK command unlocks selected entries of a specified transaction database or all transaction databases attached to a specified shared directory.

When an entry is unlocked, it can be automatically deleted when the transaction database becomes full if the transaction database has been formatted with AUTODELETE=YES or AUTODELETE=DUPS.

At least one parameter must be specified.

The following diagram illustrates the syntax for the UNLOCK command.

```
►►─UNLocK──┬─OBJect=SHRDIR,──┬─DIRDD=ddname─┬──────────────┬─,──►
           │                 └─DIRDSN=dsn───┘              │
           └─OBJect=DATABASE,──┬─DDName=ddname──────────────┬─┘
                               └─DSN=dsn,──┬─────────────┬─┘
                                           └─VOLser=volser─┘
►─┬──────────────────────┬─,─┬──────────────┬─,──►
  └─ReptNumber=xxxxxxxxxx─┘   └─Tran=xxx*───┘
►─┬──────────────┬─,─┬──────────────────┬─,──►
  └─CicsCode=xxx*─┘   └─APPLID=xxxxxxx*─┘
►─┬──────────────────────────────────────────────────────┬─,──►
  ├─DAYSOLD=nnn──────────────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,──┬──────────────────────────┬──┘
                           └─EndDATE=mm/dd/yyyy───────┘
►─┬──────────────────────────────────────────────────────┬─,──►
  │                      ┌─EndTIME=24:00:00──────┐       │
  └─StartTIME=hh:mm[:ss],──┼──────────────────────┼───────┘
                         └─EndTIME=hh:mm[:ss]────┘
  ┌─DUPS=ALL──────────┐   ┌─INC=YES──────┐   ┌─NOTFOUND=ERROR──────┐
►─┼───────────────────┼─,─┼──────────────┼─,─┼─────────────────────┼──►◄
  ├─DUPS=LOCKED───────┤   ├─INC=NO───────┤   └─NOTFOUND=IGNORE─────┘
  └─DUPS=UNLOCKED─────┘   └─INC=ONLY─────┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

Unlock entries in transaction databases attached to a shared directory that are older than six months.

```
UNLOCK OBJECT=SHRDIR,
       DIRDSN=COMPWARE.AACICS.SERVER1.SHRDIR,MONTHSOLD=6
```

VERIFY (VER)

The VERIFY command examines a selected shared directory to ensure the following:

- All transaction databases attached to the shared directory exist.
- All entries in the shared directory exist in the attached transaction databases.
- All entries in the attached transaction databases are also in the shared directory.

Note: This command creates a report of any inconsistencies but does not perform any maintenance or repairs.

The following diagram illustrates the syntax for the VERIFY command.

```
►►─VERify──┬─DIRDD=ddname─┬──────────────►◄
           └─DIRDSN=dsn───┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Transaction Databases” on page 5-5.

Example

For a shared directory, verify that all attached transaction databases exist and that the entries in the transaction database match the entries in the shared directory.

```
VERIFY DIRDSN=COMPWARE.AACICS.SERVER1.SHRDIR
```

Note: You may use the EXAMINE command to repair any inconsistencies in the Shared Directory that the VERIFY command reports. See “EXAMINE” on page 5-28 for more information about the EXAMINE command.

Chapter 6.
Batch File Utility CWAASDUT

This chapter describes the batch file utility CWAASDUT used by Abend-AID shared directories and report databases.

Notes:

Abend-AID Release 9.4 (and all subsequent releases) is the first Abend-AID release to support dump report capture to the Compuware Shared Directory.

While the file structure of the shared directory used for Abend-AID 9.4 and subsequent releases is the same as the Abend-AID for CICS shared directory, the report files created by CWAASDUT may only contain Abend-AID dump reports. When creating a report file for use by Abend-AID, the TYPE=AA utility directive must be specified; otherwise an Abend-AID for CICS report file will be created and be rendered useless for Abend-AID dump capture and reporting.

There is no software-enforced restriction in having both Abend-AID and Abend-AID for CICS report files defined to the same physical shared directory. However, Compuware does not recommend utilizing the same shared directory between products for CICS performance reasons and resultant maintenance issues.

Abend-AID dump reports may still be captured to existing DDIO report files, and this utility is not for use in creating and maintaining DDIO report files. See CWDDSUTL for maintaining DDIO report files.

Existing DDIO files from releases prior to Abend-AID 9.4 may be converted to Abend-AID shared directory report files using the CWDDCNVA Report File Conversion Utility (see **Appendix D, "Using the CWDDCNVA Report File Conversion Utility"**). A shared directory dataset must be created before converting any existing DDIO files to shared directory format. See SLCXCNTL member AACONVRT for sample JCL to convert a DDIO report file for use with the shared directory.

CWAASDUT is the only utility for use with Compuware Abend-AID Shared Directories and Report databases.

CWAASDUT must be used to delete shared directories and/or report databases to avoid possible internal corruption.

File-AID/MVS and other VSAM-related vendor products (including vendor packages that compress or enhance buffer management on VSAM files) should NOT be used against Compuware DDIO files. Doing so can corrupt the contents of the DDIO file.

CWAASDUT allows you to manage an Abend-AID shared directory and its associated abend report files. The shared directory is used to process all directory requests related to an abend report file.

For information on the CWDDSUTL utility for handling source listing files and Abend-AID report files, refer to **Chapter 4, "Batch File Utility CWDDSUTL"**.

For information on the CWFXSDUT utility for handling Abend-AID for CICS report databases and shared directories, refer to **Chapter 5, "Batch File Utility CWFXSDUT"**.

For information on the CWDDLPUT utility for handling source listing databases and shared directories, refer to **Chapter 7, "Batch File Utility CWDDLPUT"**.

The term **entry or entries** are used throughout this chapter to refer to Abend-AID diagnostic reports in a report file.

Summary of CWAASDUT Utility Commands

The following is a list and brief description of the commands available for the CWAASDUT batch file utility. Each command is discussed in detail later in this chapter.

ATTACH Connects a standalone report file to a shared directory and populates the shared directory with report file entries.

CHANGE Changes the AUTODELETE attribute of a report file.

COPY Copies entries from one shared directory or report file to another shared directory or report file.

CREATE Allocates and formats a shared directory or report file, or formats a pre-allocated shared directory or report file.

DELETE Deletes a selected shared directory or report file or selected report file entries.

DETACH Disconnects a report file from a shared directory and removes the report file entries from the shared directory.

DIRECTORY Lists selected entries in a shared directory or report file.

DIRS Lists selected directory entries in a sequential file produced by the EXPORT command.

DIRX Reports on the characteristics of a shared directory and all report files in the shared directory. Reports on the characteristics of the entries in a report file.

ESTIMATE Calculates the number of blocks to allocate for a shared directory when the allocation is being performed outside of CSS (i.e., through IDCAMS or Compuware's File-AID product).

EXAMINE Reads, analyzes, and optionally repairs the selected shared directory. Optionally deletes incomplete members in the shared directory and attached databases (if DELETEIC is specified).

EXPAND Resizes a shared directory's space allocation and/or CISIZE.

EXPORT Exports selected entries from a shared directory or report file to a sequential file, from which they can later be imported into the same or another shared directory or report file using the IMPORT command.

IMPORT Selectively restores previously exported entries from a sequential file to a shared directory or report file.

LOCK Locks the entries of a selected report file, or the entries of all report files attached to a specified shared directory, so entries cannot be automatically deleted when the report file becomes full.

MOVE Copies entries from one shared directory or report file to another shared directory or report file and deletes them from the "from" file only if they were successfully copied.

RESET Disconnects a report file from a shared directory when the shared directory is lost or damaged. The report file can then be attached to another shared directory.

SCRATCH Clears all entries from a shared directory for a specified report file that has been lost, deleted, or damaged.

UNLOCK Unlocks selected entries of a specified report file, or all report files attached to a specified shared directory, so that they can be automatically deleted when the report file becomes full.

VERIFY Ensures that the information contained in a shared directory matches the information contained in its report files. This command produces a report listing any inconsistencies but does not perform any maintenance or repairs. Use the EXAMINE command with the REPAIR directive to attempt any repair.

Usage Considerations

This section discusses special considerations for using the CWAASDUT utility.

Renaming Shared Directories and Report Files

To rename a report file, you must first detach it from the shared directory and then rename it. You must then re-attach it to the shared directory in order to retain all of the correct references. The report file, with its new name, must be connected to the shared directory before it can be used.

To rename a shared directory, you must first detach the report files from the old shared directory. Next, re-attach the report files to the renamed shared directory.

DD Statements Used with CWAASDUT

The following is a description of each of the DD statements required for CWAASDUT.

Table 6-1 contains a list of the required and optional DD statements used with CWAASDUT batch commands.

ABNLPARM

The ABNLPARM DD statement specifies the control dataset. Normally, the control dataset resides in the input stream; however, it can reside on a system input device or it can be defined as a member in a partitioned dataset. You must specify the ABNLPARM DD statement each time you run the batch file utility. The ddname ABNLPARM cannot be altered.

ABNLREPT

The ABNLREPT DD statement specifies a sequential dataset for messages and reports produced by CWAASDUT. You must specify the ABNLREPT DD statement each time you run the batch file utility. The ddname ABNLREPT cannot be altered.

ABNLFROM

The ABNLFROM DD statement specifies the input report file for the COPY, EXPORT, IMPORT, and MOVE commands. The default ddname ABNLFROM can be changed using the FROMDD parameter.

ABNLTO

The ABNLTO DD statement specifies the output report file for the COPY, EXPORT, IMPORT, and MOVE commands. The default ddname ABNLTO can be changed using the TODD parameter.

Table 6-1. DD Statements Used with CWAASDUT Batch Commands

	ABNLPARM	ABNLREPT	ABNLFROM	ABNLTO
ATTACH	X	X		
CHANGE	X	X		
COPY	X	X	O	O
CREATE	X	X		
DELETE	X	X		
DETACH	X	X		
DIR	X	X		
DIRS	X	X		
DIRX	X	X		
ESTIMATE	X	X		
EXAMINE	X	X		
EXPAND	X	X		
EXPORT	X	X	O	O
IMPORT	X	X	O	O
LOCK	X	X		
MOVE	X	X	O	O
RESET	X	X		
SCRATCH	X	X		
UNLOCK	X	X		
VERIFY	X	X		
X — Required; O — Optional				

Usage Restrictions

1. CWAASDUT is the only utility for use with Compuware Abend-AID shared directories and report files.
2. CWAASDUT must be used to delete shared directories and/or report databases to avoid possible internal corruption.
3. File-AID/MVS and other VSAM-related vendor products (including vendor packages that compress or enhance buffer management on VSAM files) should NOT be used against Compuware DDIO files. Doing so can corrupt the contents of the DDIO file.

Command Conventions

Use the following conventions when specifying commands:

- Enter commands and their parameters as control cards in the dataset defined by the ABNLPARM DD statement.

- Commands can start in any column.
- Separate the command from its parameters by one or more spaces.
- Separate parameters by commas without intervening spaces.
- When typing parameter names, type either the whole parameter name or the shortcut in parentheses in the parameters list (see "Command Parameters for Shared Directories and Report Files" on page 6-5), or the uppercase letters shown in the syntax diagram for the command. Typing a shortened version of the parameter won't work. For example, typing AUTOD for the AUTODELETE parameter is not valid — you must type AUTODELETE or AD.
- Control cards can also be continued on successive lines. Each line of a multiple-line control card statement must contain at least one parameter. To continue on a new line, enter a comma followed by a space immediately after the last parameter on the line.
- To include a comment on a line, enter at least one blank space after the last parameter on the line, followed by the comment text.
- Any line containing an asterisk (*) followed by a space in the first two columns is treated as a comment line.
- The CWAASDUT utility stops when it reaches an end-of-file (EOF) or null input control card.
- The asterisk (*) is the generic or wildcard character indicator, and it can be used in the JESID parameter:

  The wildcard character can only be used at the end of the parameter being specified.
- Parameter strings with blanks must have apostrophes around them.

Command Parameters for Shared Directories and Report Files

This section provides information on the command parameters for shared directories and report files. Not all parameters are available for all commands.

AM

Specifies the access method for the file being created. This parameter is used only when allocating or reallocating a file. Allowable values are:

VSAM: The access method is VSAM.

BSAM: The access method is sequential.

Note: A shared directory must be created as a VSAM file. Report files may be either BSAM or VSAM. VSAM is the preferred access method for Compuware access performance.

AUTODELETE (AD):

Determines the action taken when an attempt is made to write a member to a report file that is full. The AUTODELETE parameter specifies whether the oldest member in the DDIO file is automatically deleted when the file or directory becomes full. A DDIO file is considered full when all of the groups (EXTENTs) in the file have been allocated or the number of members specified by REPTCOUNT has been reached.

The autodelete process is based on **manual** and **automatic** locks. Manual locks are created online using the **L** line command from any directory screen or with the batch utility LOCK command. Automatic locks are created whenever a member is added to a DDIO file formatted using the AUTODELETE=DUPS option. The most recent member is automatically locked.

Members that have been manually locked retain their locks during the autodeletion process. They may be unlocked by using the **U** line command online or with the UNLOCK batch command. A DELETE command in batch, or the **D** line command online, will unlock a member prior to executing the delete.

For Abend-AID reports, the oldest member is the report with the earliest creation date and time. The following are valid values.

YES: (Default): Allows new members to be written to a full DDIO file by deleting the oldest unlocked members. This method is commonly referred to as **wrap-around** mode, and is typically used for DDIO report files and source listing files where members are added as needed. AUTODELETE=YES does not automatically lock members.

A file full condition will occur when AUTODELETE=YES and all members are manually locked or in use. You can resolve a file full condition by manually deleting members from the file or manually unlocking members.

Notes:

1. If you want to keep a specific member for a longer period, you can lock it, and it will be bypassed when autodelete processing selects the next report member to be automatically deleted.
2. The DELETE DUPS parameter should only be used against AUTODELETE=DUPS files. Using DELETE DUPS=UNLOCKED against an AUTODELETE=YES file will delete all unlocked members ('ALL' members).

DUPS: When a new member is added to a DDIO file, all automatically locked members with the same name are unlocked except the member with the most recent abend date and time (Abend-AID reports). Members with the same job name and abend code that have been manually locked using the **L** line command or the LOCK batch command remain locked and will not be deleted. IF THE DDIO FILE IS FULL, the oldest unlocked member not in use, regardless of name, is chosen for autodeletion.

A file full condition will occur when AUTODELETE=DUPS and all members are manually or automatically locked or in use. You can resolve a file full condition by manually deleting members from the file or manually unlocking members when AUTODELETE=DUPS.

NO: Prevents members from being written to a DDIO file when it becomes full. To add members to a full file, one or more members of the file must be manually deleted using the DELETE command.

A file full condition will occur when AUTODELETE=NO and the file is full. You can resolve a file full condition by manually deleting members from the file.

DDIO File Full Condition

A **file full condition** exists when new members cannot be added to a full DDIO file, in which case an error is returned. A DDIO file full condition will occur with the following AUTODELETE options:

YES All members are manually or automatically locked or in use.

DUPS All members are manually or automatically locked or in use.

NO The file is full.

Resolving a File Full Condition

You can perform the following steps to resolve a file full condition:

- Manually delete members from the file.
- Manually unlock members when AUTODELETE=YES or AUTODELETE=DUPS.
- Allocate and format a larger file, and use the MOVE or COPY commands to transfer the contents of the full file to the new file.
- Use the EXPORT command with the DELETE option to archive all or some members to a sequential file (tape or disk).

  Using EXPORT, space can be freed up without losing members, since members can be imported back into the file as needed. This method also avoids the overhead created by formatting new files.

BLKSIZE (BLK)

If you specified AM=BSAM, use this parameter to specify a blocksize for the sequential report file. The default is 4096 (or 4 KB). The BLKSIZE should be between 2048 (or 2 KB) and the track size. The maximum BLKSIZE is 32767.

CISIZE

If you specified AM=VSAM, use this parameter to specify the control interval size.

If neither the CISIZE nor the BLKSIZE parameter is specified, then the default CISIZE is 4096, and the default blocksize is 4089.

If the CISIZE parameter is not specified and the BLKSIZE parameter is specified, then the value for CISIZE is determined by adding 7 to the BLKSIZE value.

If the BLKSIZE parameter is not specified and the CISIZE parameter is specified, then the blocksize is determined by subtracting 7 from the CISIZE value.

If both the CISIZE and the BLKSIZE parameters are specified, then the BLKSIZE value is ignored. In this case, the blocksize is determined by subtracting 7 from the CISIZE value.

CONTINUE

Specifies whether the command will continue if recoverable errors are detected. A value of YES (default) specifies that the command will continue, if possible, when a recoverable error occurs. A value of NO specifies that the command will be stopped when an error occurs. This parameter is valid for the COPY, MOVE, EXPORT, and IMPORT commands.

CYLINDERS (CYL)

To use cylinders as the unit of measure for the size of the file, use this parameter to specify the number of cylinders.

DATABASEDISP or DBDISP

When a shared directory is deleted, this parameter determines the disposition of the report files attached to the shared directory. Allowable values are:

DETACH: (Default) Detach all attached report files from the shared directory prior to deleting the shared directory. This choice preserves the report files and allows you to attach them to another shared directory.

DELETE: Do not detach the report files from the shared directory. This choice deletes both the shared directory and the attached report files.

DATACLAS

Specifies the eight-character data class name for the Storage Management Subsystem (SMS) operating environment. This parameter is only valid for new shared directories or report files, or when the REALLOCATE parameter is set to YES.

DAYSOLD

Allows you to select entries in a shared directory or report file that are older than nnn days. For example, using the DELETE command for a report file with DAYSOLD=30 deletes all report entries that are more than 30 days old.

DDNAME (DDN)

If you have specified OBJECT=DATABASE, use this parameter for the ddname of the JCL statement that specifies the dataset name of the report file.

DELETE

Specifies whether successfully exported entries will be deleted from the "from" report file. A value of YES specifies that the entries will be deleted. A value of NO specifies that the entries will not be deleted. The default value is NO.

DELETEIC

Deletes any incomplete reports from the shared directory and report file. Incomplete reports are partial dump reports that did not complete because of any region outages or Shared Services or Abend-AID failure during dump capture. This parameter is valid only for the EXAMINE command.

DIRDD

If you have specified OBJECT=SHRDIR, this parameter indicates the ddname of the shared directory.

If you have specified OBJECT=DATABASE, this parameter indicates the ddname of the shared directory to which the report file will be attached.

DIRDSN

If you have specified OBJECT=SHRDIR, and you are **not** using a JCL statement for the dataset name, use this parameter to specify the dataset name of the shared directory. This parameter contains the actual dataset name.

If you have specified OBJECT=DATABASE, and you are **not** using the DIRDD to point to the JCL statement for the dataset name of the shared directory, you can use this parameter to specify the shared directory dataset name.

Note: You can use either the **DIRDD** or the **DIRDSN** parameter, depending on whether you are using a JCL statement for the dataset name of the shared directory.

DIRENTS

If you have specified OBJECT=SHRDIR, and you are unable to determine the number of blocks you will need for the shared directory, you can use this parameter to allocate an appropriate number of blocks. This parameter specifies the total number of directory entries expected in the shared directory. The total number of directory entries consists of the total number of Abend-AID reports for each report file attached to the shared directory. When combined with a blocksize, the DIRENTS parameter is used by CWAASDUT to calculate a reasonable number of blocks for allocation. The blocksize can be specified directly in the BLKSIZE parameter or indirectly through the CISIZE or MODEL parameter.

If you are using an external method to allocate blocks (e.g., IDCAMS or Compuware's File-AID), you can use the ESTIMATE command to determine the number of blocks you will need. See "ESTIMATE" on page 6-25 for more information.

DIRVOL

If you have specified OBJECT=SHRDIR, and you are **not** using a JCL statement for the volume serial number of the shared directory, you can use this parameter to specify the volume serial number.

DSN

If you have specified OBJECT=DATABASE, and you are **not** using a JCL statement for the dataset name, use this parameter to specify the dataset name of the report file.

DUPS

Specifies whether all entries, only locked entries, or only unlocked entries are selected. The available values are: ALL (default), LOCKED, or UNLOCKED.

ENDDATE (EDATE)

Specifies the last processing date of the Abend-AID reports. This parameter can be specified as either MM/DD/YY or MM/DD/YYYY. Acceptable four-digit years are 1970 through 2069. If ENDDATE is specified, then STARTDATE must also be specified. The highest valid ENDDATE is 12/31/2069.

ENDTIME (ETIME)

Specifies the last processing time of the affected reports. This parameter must be specified as HH:MM[:SS] or HH.MM[.SS]. If seconds are omitted, the default is 59.

This parameter must be preceded by the STARTTIME parameter. If the STARTTIME parameter is specified and the ENDTIME parameter is omitted, then ENDTIME defaults to 24:00:00.

When using STARTTIME and ENDTIME in conjunction with STARTDATE and ENDDATE, entries are first selected by date, then by time.

EXTENTS

Specifies the maximum number of extents to be allocated to each report file entry. If the EXTENTS parameter is not specified, the default value is 64. The maximum value is 460.

FROMDD

Specifies the ddname of the report file or shared directory from which the entries are copied, exported, or moved. It also specifies the ddname of the sequential file from which previously exported entries are imported.

FROMDSN

Specifies the DSN of the report file or shared directory from which the entries are copied, exported, or moved. It also specifies the DSN of the sequential file from which previously exported entries are imported.

FROMOBJECT

Specifies the type of file from which the entries are copied, exported, or moved. Allowable values are:

SHRDIR: The “from” file is a shared directory.

DATABASE: The “from” file is a report file.

FROMVOL

Specifies the VOLSER of the report file or shared directory from which the entries are copied, exported, or moved. It also specifies the VOLSER of the sequential file from which previously exported entries are imported.

GROUPCOUNT (GC)

Specifies the number of blocks allocated to a report file entry each time additional space is needed. The minimum value is 1, and the maximum value is 2048. The default value is 4.

INC

Specifies whether to include incomplete entries in the selection. Allowable values are:

YES: (Default) Select both complete and incomplete entries.

NO: Select only complete entries.

ONLY: Select only incomplete entries.

JESID

Specifies reports by JESID. Wildcard characters are valid.

```
JESID=J12345 | T12345 | S12345
```

JOBNAME

Specifies abend reports by the JOB name. Wildcard characters are valid.

```
JOBNAME=ACCNTNG  JOBNAME=A*
```

LABEL

Specifies the file number for the sequential dataset on the tape for the IMPORT, EXPORT, and DIRS commands. (VOL=SER=, and other parameters should be specified on the corresponding DD statement). The default value is 1.

The LABEL parameter allows you to reference multiple export files using a single ddname. It also allows you to create multiple export files on a single tape. For example, to include a report file on a tape that already contains the related source code and compiler output as files 1 and 2, you can specify LABEL=3 in the EXPORT command for the report file. This will append the report file on the tape as file 3. Later, to perform a DIRS or IMPORT on that file, include LABEL=3 with the command.

The LABEL parameter is valid for IMPORT, EXPORT, and DIRS. You can include the LABEL parameter in your JCL.

Note: If you include the LABEL parameter in your JCL, but have also specified a LABEL parameter with the IMPORT or EXPORT batch command, the LABEL parameter in the JCL will be overridden.

For compatibility with the IBM Tape Management System, the label number ".Lnnn" is appended to the tape DSNAME as the last qualifier. If the combined length of the specified DSNAME and the appended label number exceeds the maximum length for DSNAME, an error message is issued.

MGMTCLAS

Specifies the eight-character management class name for the Storage Management Subsystem (SMS) operating environment. This parameter is only valid for new shared directories or report files, or when the REALLOCATE parameter is set to YES.

MODEL

Use this parameter to specify the DSN of an existing shared directory or report file that can be used as a "template" for creating a new shared directory or report file with the same VSAM characteristics.

When the MODEL parameter is used, you can override any of the VSAM or BSAM parameters by including the appropriate parameter with the CREATE or EXPAND command. For example, to use a particular shared directory as a template for creating

a new shared directory, but with a different CISIZE, include the CISIZE parameter in the CREATE command.

MONTHSOLD

Allows you to select entries in a shared directory or report file that are older than nnn months. For example, using the DELETE command for a report file with MONTHSOLD=6 deletes all report entries older than six months.

NOTFOUND

Specifies whether to request a return code 00, instead of the default code 44, when an entry is not found. The valid values are:

ERROR: (Default) Allows the default code.

IGNORE: Forces a return code 00.

OBJECT (OBJ)

Specifies the type of object being changed, deleted, locked, or unlocked; or the type of object being created, reallocated, or formatted. This parameter also specifies the shared directory for the EXPAND command, and the shared directory or the report file for the DIRX command. Allowable values are:

SHRDIR: Shared directory

DATABASE: Abend-AID report file

An additional option available for the DELETE command is:

ENTRY: Member in the file. This option is only used with the DELETE command when removing an entry from a report file and the shared directory to which it is attached.

REALLOCATE (REALLO)

If the shared directory or report file you have specified already exists, you can use this parameter to reallocate the file. Allowable values are:

NO: (Default) For sequential files, the file will be reformatted if possible. For VSAM files, an error message is returned and the file is left unchanged.

YES: For both VSAM and sequential files, the file will be deleted and then reallocated.

RECORDS (RECS)

If you have specified AM=VSAM and want to use records as the unit of measure for the size of the file, use this parameter to specify the number of records. This parameter is valid for VSAM files only.

REPAIR

Attempts repair of any shared directory file structure errors and directory entry mismatches between the shared directory and all of its attached report databases.

REPTCOUNT (RC)

Specifies the maximum number of entries in the report file, provided there is enough space. The minimum value is 1, and the maximum number of Abend-AID diagnostic reports that can exist in a report file is 16380. The default is 100.

REPTNUMBER (RN)

Specifies an entry's number. If this parameter is specified, all other selective parameters are ignored.

RETAINRN

Specifies whether the entry's report number will be retained during a move or copy. Allowable values are:

NO (Default) Do not retain the number.

YES: Retain the number. In this case, if the number to be retained is already being used in the "to" file, then the next highest available number will be used. This value should only be used when moving or copying to a shared directory.

SECONDARY

This parameter can only be used when formatting pre-allocated shared directories or report files. It allows shared directories or VSAM report files to be allocated with secondary extents. Allocating secondary extents allows a shared directory or VSAM report file to be allocated on multiple volumes. The default is NO.

Note: This parameter is only valid for shared directories or VSAM report files, and the VSAM cluster must be pre-allocated.

SORT

Specifies the order in which entries are selected for the execution of a command. By default, entries are selected in descending date/time order:.

- Entries are first ordered by date, from newest (most recent) to oldest.
- Entries with the same date are then ordered by time (from latest to earliest).

Valid sort keys, their abbreviations, and the sorting sequence they cause are specified in Table 6-2.

Table 6-2. Report File Sort keys for Abend-AID

Sort Key	Abbreviation	Primary Sort Field	Sequence	Secondary Sort Field	Sequence	Tertiary Sort Field	Sequence
JOBNAME		JOBNAME	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
NUMBER	NUM	REPT-NUMBER	Descending				
DATE		DATE/TIME	Descending	REPT-NUMBER	Descending		

STARTDATE (SDATE)

Specifies the processing date of the reports. The STARTDATE parameter is used with the ENDDATE parameter to specify a date range.

The STARTDATE parameter can be specified as either MM/DD/YY or MM/DD/YYYY. Acceptable four-digit years are 1970 through 2069. When this parameter is used alone, all entries for the specified date are selected.

The earliest valid start date is 01/01/1970.

STARTTIME (STIME)

Specifies the beginning processing time of the reports. This parameter must be specified as HH:MM[:SS] or HH.MM[.SS]. If seconds are omitted, the default is 00. This parameter must precede the ENDTIME parameter (if ENDTIME is specified).

If the STARTTIME parameter is specified and the ENDTIME parameter is omitted, then ENDTIME defaults to 24:00:00.

When using STARTTIME and ENDTIME in conjunction with STARTDATE and ENDDATE, entries are first selected by date, then by time.

STORCLAS

Specifies the eight-character storage class name for the Storage Management Subsystem (SMS) operating environment. This parameter is only valid for new shared directories or report files, or when the REALLOCATE parameter is set to YES.

TODD

Specifies the ddname of the report file or shared directory to which the entries are copied, exported, or moved. It also specifies the ddname of the sequential file to which entries are exported.

TODSN

Specifies the DSN of the report file or shared directory to which the entries are copied, exported, or moved. It also specifies the DSN of the sequential file to which entries are exported.

TOOBJECT

Specifies the type of file to which the entries are copied, imported, or moved. Allowable values are:

SHRDIR: The “to” file is a shared directory.

DATABASE: The “to” file is an Abend-AID report file.

TOVOL

Specifies the VOLSER of the report file or shared directory to which the entries are copied, exported, imported, or moved. It also specifies the VOLSER of the sequential file to which previously exported entries are exported.

TRACKS (TRKS)

To use tracks as the unit of measure for the size of the file, use this parameter to specify the number of tracks. This parameter is valid for both VSAM and sequential files.

TYPE

If you are creating a report file, use this parameter to specify the type of entries the file can contain. For an Abend-AID report file, code TYPE=AA.

UNIT

If you have specified AM=BSAM, you can use this parameter to specify a device or device type. The default is SYSDA.

VOLSER (VOL)

If you have specified OBJECT=DATABASE, you can use this parameter to specify the volume serial number of the report file when you are using the DSN parameter.

VSAMCAT

If you have specified AM=VSAM, you can use this parameter to specify the VSAM catalog in which the file will be defined. This is an optional parameter.

YEARSOLD

Allows you to select entries in a shared directory or report file that are older than nnn years. For example, using the DELETE command for a report file with YEARSOLD=1 deletes all Abend-AID report entries that are older than one year.

Commands Descriptions and Syntax

This section describes the commands used to manage shared directories and report files including valid command abbreviations and the parameters allowed for each command. With the exception of specifying REPTNUMBER, you can enter more than one parameter with a command.

ATTACH (ATT)

The ATTACH command connects a report file to a shared directory and populates the shared directory with report file entries. A report file must be attached to a shared directory before it can be used.

The following diagram illustrates the syntax for the ATTACH command.

```
►►─ATTach──┬─DIRDD=ddname─┬─,─┬─DDName=ddname─┬──►◄
           └─DIRDSN=dsn───┘   └─DSN=dsn────────┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Attach a report file to a shared directory.

```
ATTACH DSN=COMPWARE.AA.DB01,
       DIRDSN=COMPWARE.AA.SHRDIR
```

Note: If an error occurs during the ATTACH process, when adding members into the shared directory, the database will set “attach in progress” (will show on a DIRX), and the shared directory MAY contain entries for only some (but not all) of the database members. If the problem cannot be solved by another attempt to ATTACH, then perform the following to eliminate possible integrity problems:

Run DETACH on the database to clear the ATTACH setting and remove entries added during the ATTACH process.

CHANGE (CHG)

The CHANGE command changes the AUTODELETE attribute of a report file.

The following diagram illustrates the syntax for the CHANGE command.

```
►►─CHanGe──OBJect=DATABASE,─┬─DDName=ddname──────────────┬─,──►
                            └─DSN=dsn,─┬──────────────┬─┘
                                       └─VOLser=volser─┘

  ┌─AutoDelete=YES─┐
►─┼────────────────┼──►◄
  ├─AutoDelete=DUPS─┤
  └─AutoDelete=NO──┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Change the AUTODELETE attribute on a report file from AUTODELETE=NO to AUTODELETE=DUPS.

```
CHANGE OBJECT=DATABASE,DSN=COMPWARE.AA.DB01,AD=DUPS
```

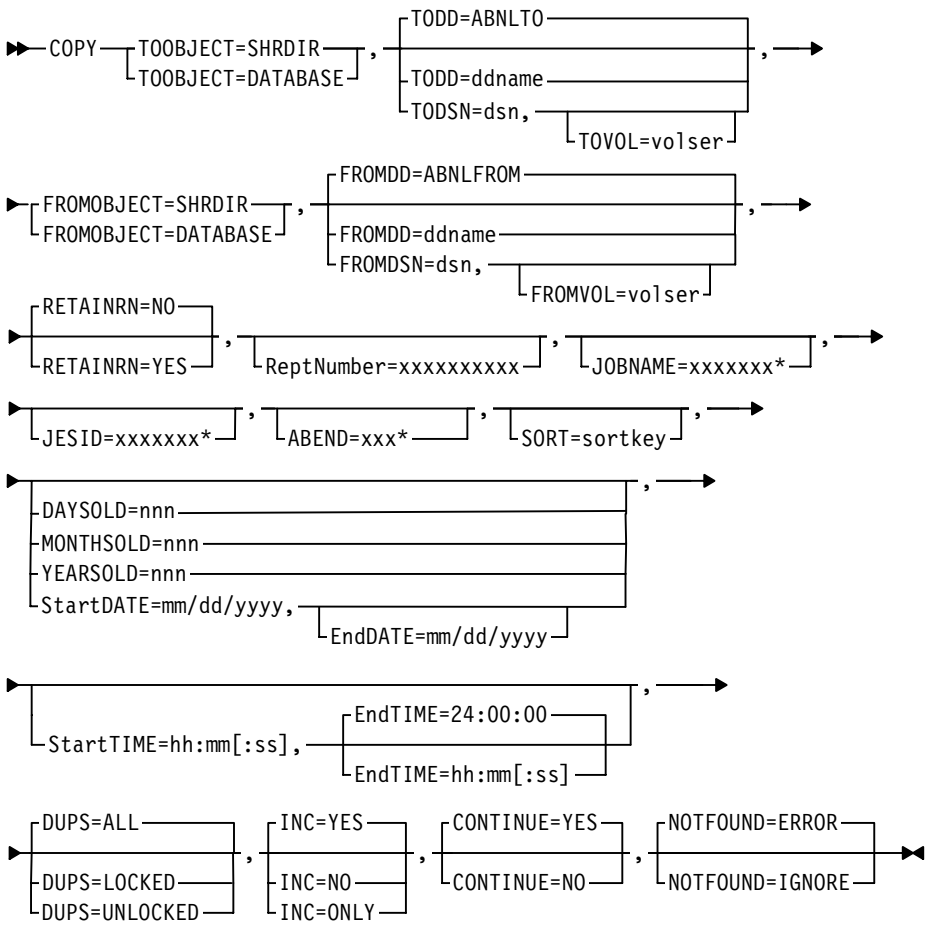
COPY

The COPY command copies selected entries in any of the following ways:

- from one report file to another report file
- from a report file to a shared directory
- from a shared directory to a report file
- from one shared directory to another shared directory.

Use the FROMDD and TODD parameters to specify the “from” and “to” file ddnames. When an entry is copied to or from a report file that is attached to a shared directory, the shared directory is also updated immediately. If no other parameters are specified, then all entries are copied.

The following diagram illustrates the syntax for the COPY command.



For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Copy all entries from a report file to another report file.

```
COPY TOOBJECT=DATABASE,TODSN=COMPWARE.AA.DB01,
     FROMOBJECT=DATABASE,FROMDSN=COMPWARE.AA.DB02
```

CREATE (CRE)

The CREATE command performs the following functions:

- Allocates and formats a new shared directory
- Formats a pre-allocated shared directory
- Allocates and formats a new report file.
- Formats a pre-allocated report file.

CAUTION:
This command erases all of the information in the specified file.

Only VSAM files can be used for shared directory files; however, either VSAM or sequential files can be used for report files. The preferred access method is VSAM.

When you create a report file, it is automatically connected to a shared directory that is specified as the DIRDD or DIRDSN parameter of the CREATE command. A report file must be connected to a shared directory before it can be used for dump capture.

Sample JCL Members

Sample JCL members for the CREATE command are available in CSS installation library members. Please refer to Table 6-3 for a list of the sample members.

Table 6-3. Sample JCL Members for CREATE Command

To format...	Use sample member...
VSAM Report Files	CXALLVAA
Shared Directories	CXALLDAA

The following diagrams illustrate the syntax for the CREATE command based on the function you want to perform.

Allocating and Formatting a New Shared Directory:

The first CREATE syntax diagram illustrates the syntax for allocating and formatting a new shared directory.

```
►►─CREate─OBJect=SHRDIR,──┬─DIRDD=ddname──────────────────┬─,──►
                          └─DIRDSN=dsn,─┬──────────────┬─┘
                                        └─DIRVOL=volser─┘

►─AM=VSAM,──┬──────────────┬─,──┬──────────────────┬─,──►
            ├─RECords=nnnn─┤    │ ┌─CISIZE=4096─┐   │
            ├─TRacKS=nnnn──┤    ├─┼─────────────┼───┤
            ├─CYLinderS=nnnn─┤    │ └─CISIZE=nnnn─┘   │
            └─DIRENTS=nnnn─┘    │ ┌─BLKsize=4089─┐   │
                                └─┼──────────────┼───┘
                                  └─BLKsize=nnnn─┘

                                    ┌─REALLOcate=NO─┐
►─┬──────────────┬─,─┬──────────┬─,─┼───────────────┼─,──►
  └─VSAMCAT=ddname─┘  └─MODEL=dsn─┘  └─REALLOcate=YES─┘

►─┬──────────────────┬─,─┬──────────────────┬─,─┬──────────────────┬──►◄
  └─MGMTCLAS=xxxxxxxx─┘  └─STORCLAS=xxxxxxxx─┘  └─DATACLAS=xxxxxxxx─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

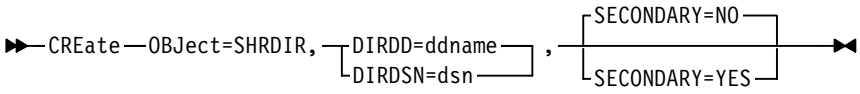
Example

Create a shared directory with a CISIZE of 18425, which can occupy five cylinders.

```
CREATE OBJECT=SHRDIR,DIRDSN=COMPWARE.AA.SHRDIR,
       AM=VSAM,CYLINDERS=5,CISIZE=18425,DIRVOL=ABCDEF
```

Formatting a Pre-Allocated Shared Directory

This CREATE syntax diagram illustrates the syntax for a pre-allocated shared directory.



For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

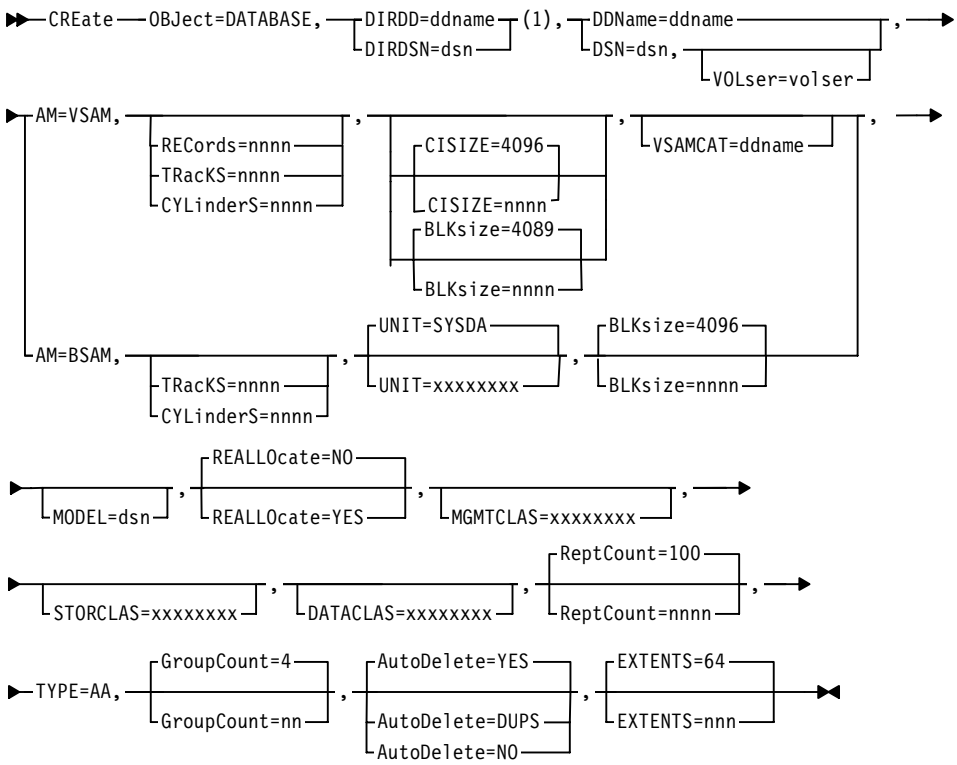
Example

Format a pre-allocated shared directory and specify use of secondary extents.

```
CREATE OBJECT=SHRDIR,
       DIRDSN=COMPWARE.AA.SHRDIR2,
       SECONDARY=YES
```

Allocating and Formatting a New Report File

This CREATE syntax diagram illustrates the syntax for allocating and formatting a new report file.



Note:

1. Please note that the DIRDD parameter (or DIRDSN) specifies the name of the shared directory to which you are attaching the report file. The DDNAME parameter (DSN) specifies the name of the report file.

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Create a VSAM report file with the following characteristics:

- The maximum number of entries in the file is 50.
- At least one entry of each name will remain in the report file.
- When additional space is required in the report file, three blocks will be allocated.
- The maximum number of extents to be allocated to each report file entry is 256.
- The report file can occupy 5 cylinders.
- The blocksize is 18432.

```
CREATE OBJECT=DATABASE,TYPE=AA,DIRDSN=COMPWARE.AA.SHRDIR,
       DSN=COMPWARE.AA.DB01,
       RC=50,AD=DUPS,GC=3,EXTENTS=256,AM=VSAM,CYLINDERS=5,CISIZE=18432
```

Formatting a Pre-Allocated Report File

This CREATE syntax diagram illustrates the syntax for formatting a pre-allocated report file.

```
►►─CREate─OBJect=DATABASE, ─┬─DIRDD=ddname─┬─, ─┬─DDName=ddname─┬─, ──►
                            └─DIRDSN=dsn───┘    └─DSN=dsn───────┘

   ┌─SECONDARY=NO──┐   ┌─ReptCount=100──┐                ┌─GroupCount=4──┐
►─┼───────────────┼─, ┼───────────────┼─, ─TYPE=AA, ─┼───────────────┼─, ──►
   └─SECONDARY=YES─┘   └─ReptCount=nnn──┘                └─GroupCount=nn──┘

   ┌─AutoDelete=YES──┐   ┌─EXTENTS=64──┐
►─┼─────────────────┼─, ┼─────────────┼──►◄
   ├─AutoDelete=DUPS─┤   └─EXTENTS=nnn──┘
   └─AutoDelete=NO───┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Format a pre-allocated report file with the following characteristics:

- The maximum number of entries in the file is 100.
- Autodelete entries when full.
- When additional space is required in the report file, two blocks will be allocated.
- The maximum number of extents to be allocated to each report file entry is 64.

```
CREATE OBJECT=DATABASE,
       TYPE=AA,
       DIRDSN=COMPWARE.AA.SHRDIR,
       DSN=COMPWARE.AA.DB01,
       RC=100,GC=2,AD=YES,EXTENTS=64
```

DELETE (DEL)

The DELETE command deletes selected report file entries, a selected report file, or a selected shared directory.

CAUTION:

To delete reports from the Shared Directory and associated report file, use: OBJECT=ENTRY. Deletion may be specific using RN(REPTNUMBER) or non-specific using JOBNAME= STARTDATE, ENDDATE, or DAYSOLD, MONTHSOLD, or YEARSOLD.

DELETE an ENTRY from Both the Shared Directory and Report File

DELETE removes the entry from both the shared directory and the report file. report file entries can only be deleted if attached to a shared directory. You must specify either the DIRDD or the DIRDSN parameter.

Be certain to specify OBJECT=ENTRY to delete reports from the shared directory and associated report file.

```
▶▶─DELete──────OBJect=ENTRY,──┬─DIRDD=ddname─┬─,──┬──────────────────────┬─,──▶
                              └─DIRDSN=dsn───┘    └─ReptNumber=xxxxxxxxxx─┘

▶──┬──────────────────┬─,──┬──────────────┬─,──┬──────────────┬─,──▶
   └─JOBNAME=xxxxxxx*─┘    └─ABEND=xxx*────┘    └─JESID=xxxxxxx*─┘

▶──┬─DAYSOLD=nnn──────────────────────────────────────────────┬─,──▶
   ├─MONTHSOLD=nnn─────────────────────────────────────────────┤
   ├─YEARSOLD=nnn──────────────────────────────────────────────┤
   └─StartDATE=mm/dd/yyyy,──┬──────────────────────────────────┘
                            └─EndDATE=mm/dd/yyyy─┘

                                                   ┌─DUPS=ALL────────┐
▶──┬──────────────────────────────────────────────┬─┴─,─┬─────────────┴─,──▶
   │                                              │     ├─DUPS=LOCKED─┤
   └─StartTIME=hh:mm[:ss],──┬─EndTIME=24:00:00─┬─┘     └─DUPS=UNLOCKED─┘
                            └─EndTIME=hh:mm[:ss]─┘

   ┌─INC=YES──┐   ┌─NOTFOUND=ERROR──┐
▶──┴──────────┴─,─┴──────────────────┴──▶◀
   ├─INC=NO───┤   └─NOTFOUND=IGNORE──┘
   └─INC=ONLY─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Delete all entries with a JOBNAME of DAILYRUN.

```
DELETE OBJECT=ENTRY,
       DIRDSN=COMPWARE.AA.SHRDIR,
       JOBNAME=DAILYRUN
```

DELETE a Report File (and Detach It from the Shared Directory)

If the report file is attached to a shared directory, DELETE also detaches it from the shared directory prior to deletion. You must specify either the DDNAME or the DSN parameter. (To delete a report file, specify OBJECT=DATABASE.)

```
►►─DELete──OBJect=DATABASE,─┬─DDName=ddname──────────────┬──►◄
                            └─DSN=dsn,─┬──────────────┬─┘
                                       └─VOLser=volser─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Delete a report file (and detach it from the shared directory).

```
DELETE OBJECT=DATABASE,
       DSN=COMPWARE.AA.SHRDIR01
```

DELETE a Shared Directory

When deleting a shared directory, you can choose whether to also delete all attached report files. Regardless of your choice, a DETACH is performed on all attached report files prior to the deletion. You must specify either the DIRDD or the DIRDSN parameter. (To delete a shared directory, specify OBJECT=SHRDIR.)

```
                                          ┌─DataBaseDISP=DETACH─┐
►►─DELete─OBJect=SHRDIR,─┬─DIRDD=ddname─┬─,─┼─────────────────────┼──►◄
                         └─DIRDSN=dsn───┘   └─DataBaseDISP=DELETE─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Examples

Delete a shared directory and also delete all report files attached to it.

```
DELETE OBJECT=SHRDIR,
       DIRDSN=COMPWARE.AA.SHRDIR,
       DBDISP=DELETE
```

Delete a shared directory, but first detaches any report files attached to it (does NOT delete databases).

```
DELETE OBJECT=SHRDIR,
       DSN=COMPWARE.AA.SHRDIR,
       DBDISP=DETACH
```

DETACH

The DETACH command disconnects a report file from a shared directory and removes the report file entries from the shared directory.

To ensure that you detach the report file from the correct shared directory, you may optionally use the DIRDSN parameter to specify the name of the shared directory. If the shared directory is specified incorrectly, an error will be returned. If the DIRDSN is not specified, then the report file will be unconditionally detached from the appropriate shared directory.

The following diagram illustrates the syntax for the DETACH command.

```
>>-DETACH---+-DDName=ddname-------------+-,-+-------------+-->
            +-DSN=dsn,-+-------------+-+   +-DIRDSN=dsn-+
                       +-VOLser=volser-+
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Detach a report file from a shared directory.

```
DETACH DDNAME=TRFILE1
```

Note: If an error occurs during the DETACH process, when deleting members from the shared directory, the database will set “detach in progress” (will show on a DIRX) but will still appear attached, and the shared directory may still contain entries for members on the database that had not yet been removed from the shared directory. To eliminate possible integrity problems perform the following:

1. Run RESET on the database to clear the attach setting.
2. Run SCRATCH on the shared directory for that database to remove any entries left for that database.

DIRECTORY (DIR, D) and DIRX (DX)

The DIRECTORY command lists selected entries in a shared directory or report file.

The DIRX command functions similarly to the DIRECTORY command, but it also displays attribute and allocation information. When DIRX is performed on a shared directory, it displays the attribute and allocation information for the shared directory and all report files attached to that shared directory.

The following diagram illustrates the syntax for the DIRECTORY or DIRX command.

```
►►─DIRectory or DIRX─┬──────────────────────┬─,──►
                     └ReptNumber=xxxxxxxxxx─┘

►─┬─OBJect=SHRDIR,──┬DIRDD=ddname─┬──────────────┬─,──►
  │                 └DIRDSN=dsn──┘              │
  └─OBJect=DATABASE,──┬DDName-ddname──────────────┤
                      └DSN=dsn,─┬──────────────┬─┘
                                └VOLser=volser─┘

►─┬──────────────────┬─,─┬──────────────┬─,─┬──────────────────┬─,──►
  └JOBNAME=xxxxxxx*─┘   └ABEND=xxx*───┘   └JESID=xxxxxxx*─┘

►─┬──────────────────────────────────────────────────────┬─,──►
  ├DAYSOLD=nnn──────────────────────────────────────────┤
  ├MONTHSOLD=nnn────────────────────────────────────────┤
  ├YEARSOLD=nnn─────────────────────────────────────────┤
  └StartDATE=mm/dd/yyyy,──┬──────────────────────┬──────┘
                          └EndDATE=mm/dd/yyyy─┘

►─┬──────────────────────────────────────────────────────┬─,──►
  │                      ┌EndTIME=24:00:00──────┐        │
  └StartTIME=hh:mm[:ss],─┼──────────────────────┼────────┘
                         └EndTIME=hh:mm[:ss]───┘

  ┌DUPS=ALL──────┐   ┌INC=YES──┐   ┌NOTFOUND=ERROR──────┐
►─┼──────────────┼─,─┼─────────┼─,─┼────────────────────┼─,──►
  ├DUPS=LOCKED───┤   ├INC=NO───┤   └NOTFOUND=IGNORE───┘
  └DUPS=UNLOCKED─┘   └INC=ONLY─┘

►─┬──────────────┬──►◄
  └SORT=sortkey─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5. Refer to Shared Services SLCXCNTL installation library member CXAADIRX for sample JCL for the DIRX command.

Example 1

List all entries in a shared directory.

```
DIR OBJECT=SHRDIR,DIRDSN=COMPWARE.AA.SHRDIR
```

Example 2

List all entries in a report file sorted by JOBNAME, and include attribute and allocation information.

```
DIRX OBJECT=DATABASE,DSN=COMPWARE.AA.RPT01,SORT=JOBNAME
```

DIRECTORY SEQUENTIAL (DIRS)

The DIRS (DIRectory Sequential) command lists selected directory entries in a sequential file produced by the EXPORT command.

If no other parameters are specified, then all entries are listed.

The following diagram illustrates the syntax for the DIRECTORY SEQUENTIAL command.

```
>>-DIRS---+-DDName=ddname-------------------+-,-->
          +-DSN=dsn,-+-------------------+-+
                     +-VOLser=volser-----+

>-+-------------------------+-,-+-------------------+-,-+-------------------+-,-->
  +-ReptNumber=xxxxxxxxxx-+   +-JOBNAME=xxxxxxx*-+   +-ABEND=xxx*-+

>-+-------------------------------------------------------------+-,-->
  +-DAYSOLD=nnn-------------------------------------------------+
  +-MONTHSOLD=nnn-----------------------------------------------+
  +-YEARSOLD=nnn------------------------------------------------+
  +-StartDATE=mm/dd/yyyy,-+-------------------------+-----------+
                          +-EndDATE=mm/dd/yyyy-+

>-+-------------------------------------------------------------+-,-->
  +-StartTIME=hh:mm[:ss],-+-EndTIME=24:00:00-----+-------------+
                          +-EndTIME=hh:mm[:ss]-+

  +-DUPS=ALL-------+   +-INC=YES-----+   +-NOTFOUND=ERROR-----+
>-+-------------------+-,-+-------------+-,-+-------------------+-,-->
  +-DUPS=LOCKED----+   +-INC=NO------+   +-NOTFOUND=IGNORE----+
  +-DUPS=UNLOCKED--+   +-INC=ONLY----+

                       +-LABEL=1-----+
>-+-------------------+-,-+-------------+-->
  +-SORT=sortkey----+   +-LABEL=nn----+
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Export all unlocked entries and list all directory entries on the resulting sequential file.

```
EXPORT DUPS=UNLOCKED
DIRS DSN=EFHBRR0.EXFILE1
```

ESTIMATE

The ESTIMATE command calculates a reasonable number of blocks to allocate for a shared directory when the allocation is performed outside of CSS (i.e., through IDCAMS or Compuware's File-AID product). The resulting calculation is expressed in records.

The following diagram illustrates the syntax for the ESTIMATE command.

```
►►─ESTIMATE─OBJect=SHRDIR, ─┬─BLKsize=nnnn─┬─,─┬──────────────┬─►◄
                            ├─CISIZE=nnnn──┤   └─DIRENTS=nnnn─┘
                            └─MODEL=dsn────┘
```

For a description of the parameters see "Command Parameters for Shared Directories and Report Files" on page 6-5.

Example

Determine the block allocation for a shared directory with a maximum of 200 entries and a CISIZE of 18432.

```
ESTIMATE OBJECT=SHRDIR,CISIZE=18432,DIRENTS=200
```

EXAMINE

The EXAMINE command reads, analyzes, and **optionally** repairs the selected shared directory to ensure the following:

- Internal shared directory structure integrity. Shared directory index and data components match. That is, all entries in the shared directory are accessible via an indexed search, and are selectable for abend diagnosis viewing.
- All report files attached to the shared directory still exist in the MVS catalog.
- All entries in the shared directory exist in the attached report files and can be opened for abend diagnosis.

Note: This command creates a report of any inconsistencies and **optionally with the REPAIR directive specified**:

- repairs shared directory index pointers,
- deletes from the shared directory orphaned data entries that are not selectable via the indexed path,
- deletes members from the shared directory that no longer exist in the attached database files
- deletes members from the attached databases, that are no longer in the shared directory
- deletes incomplete members in the shared directory and attached databases (if DELETEIC specified).

The following diagram illustrates the syntax for the EXAMINE command.

```
►►--EXAMINE--┬DIRDD=ddname---┬,┬---------┬,┬---------┬--►◄
             └DIRDSN=dsn-----┘ └REPAIR---┘ └DELETEIC--┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Note: The REPAIR directive attempts exclusive ENQ control of the selected shared directory.

EXAMINE may be run any time, **without** the REPAIR directive, to analyze the shared directory’s integrity. However, during periods of dump capture activity, INDEX/DATA mismatches may be erroneously reported on the EXAMINE report.

An effort was made to prevent EXAMINE/REPAIR from rendering the shared directory or its contents, unusable or impaired after REPAIR processing. However, a backup of the shared directory before using REPAIR is advised.

Runtime Example

```
EXAMINE DIRDSN=COMPUWARE.AA.SHRDIR1
```

EXPAND

The EXPAND command reallocates an existing shared directory using new allocation parameters and copies all of the entries to the newly allocated shared directory. After the copy has completed successfully, the old shared directory is renamed to a temporary name, the new shared directory is renamed to the original name, and the old shared directory is then deleted. You can use this command to increase or decrease the space allocated for a shared directory and/or change the CISIZE.

Note: If you are receiving "TARGET DATASET FULL" during abend dump capture, you may need to create additional dump report files. The size of the shared directory is not likely to be the cause.

When using the EXPAND command, the following rules apply:

- The shared directory must already exist (i.e., you cannot use EXPAND to create a new shared directory).
- The shared directory must not be in use.

Use the DIRDD or DIRDSN parameter to specify the existing shared directory. You can optionally use the VOLSER to specify the desired volume location for the new expanded shared directory.

The following diagram illustrates the syntax for the EXPAND command.

```
►►─EXPAND─OBJect=SHRDIR,─┬─DIRDD=ddname─┬─,─┬──────────────┬─,──►
                         └─DIRDSN=dsn───┘   └─VOLser=volume─┘

                                                ┌─CISIZE=4096─┐
►─┬──────────────────┬─,─┬──────────────┬─,─┼──────────────┼─,─┬──────────────┬─►◄
  └─VSAMCAT=ddname─┘   ├─RECords=nnnn──┤   └─CISIZE=nnnn─┘   └─MODEL=dsn─┘
                       ├─TRacKS=nnnn───┤
                       └─CYLinders=nnnn┘
```

For a description of the parameters see "Command Parameters for Shared Directories and Report Files" on page 6-5.

Example

Expand an existing shared directory, but change the CISIZE to 18432.

```
EXPAND OBJECT=SHRDIR,DIRDD=CAT11,
       MODEL=COMPWARE.AA.SHRDIR,
       CISIZE=18432
```

EXPORT

The EXPORT command exports selected entries from a shared directory or report file to a sequential file, from which they can later be imported into the same or another shared directory or report file using the IMPORT command.

The “from” file (the shared directory or report file) can be specified using either the FROMDSN or FROMDD parameter.

The “to” file (the sequential file) can be specified using either the TODSN and TOVOL parameters, or by using the TODD parameter. However, if you are exporting to a tape, you **must** use the TODD parameter and not the TODSN and TOVOL parameters. (The ABNLFROM and ABNLTO DD statements may be used for exporting.) When an entry is exported from a report file and DELETE=YES is specified, the shared directory is also updated.

If no other parameters are specified, then all entries are exported.

You may use multiple EXPORT commands if you include a unique TODD (or TODSN and TOVOL) parameter for each output dataset or PDS member that is a target of the export. If unique parameters are not used, each successive export will overlay the previous export.

The following diagram illustrates the syntax for the EXPORT command.

```
►►─EXPORT──┬─FROMOBJECT=SHRDIR──┬─,──────►
           └─FROMOBJECT=DATABASE─┘

►─┬─FROMDD=ABNLFROM──────────────┬─,─┬─TODD=ABNLTO──────────────┬─,──►
  ├─FROMDD=ddname─────────────────┤   ├─TODD=ddname─────────────┤
  └─FROMDSN=dsn,──┬─────────────┬─┘   └─TODSN=dsn,──┬─────────────┬─┘
                  └─FROMVOL=volser┘                 └─TOVOL=volser┘

►─┬──────────────────────┬─,─┬──────────────────────┬─,─►
  └─ReptNumber=xxxxxxxxxx─┘   └─JOBNAME=xxxxxxx*─┘

►─┬──────────────┬─,─┬──────────────┬─,──────►
  └─ABEND=xxx*───┘   └─JESID=xxxxxxx*─┘

►─┬──────────────────────────────────────────────────┬─,──►
  ├─DAYSOLD=nnn──────────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,─┬─────────────────────┬─┘
                          └─EndDATE=mm/dd/yyyy─┘

►─┬──────────────────────────────────────────────────┬─,──►
  └─StartTIME=hh:mm[:ss],─┬─EndTIME=24:00:00──┬─┘
                          └─EndTIME=hh:mm[:ss]─┘

►─┬─DUPS=ALL──────┬─,─┬─INC=YES──┬─,─┬─NOTFOUND=ERROR──┬─,──►
  ├─DUPS=LOCKED───┤   ├─INC=NO───┤   └─NOTFOUND=IGNORE─┘
  └─DUPS=UNLOCKED─┘   └─INC=ONLY─┘

►─┬─CONTINUE=YES─┬─,─┬─LABEL=1──┬─,─┬─DELETE=NO─┬──►◄
  └─CONTINUE=NO──┘   └─LABEL=nn─┘   └─DELETE=YES─┘
```

For a description of the parameters see "Command Parameters for Shared Directories and Report Files" on page 6-5. Refer to Shared Services SLCXCNTL installation library member CXAAEXPO for sample JCL for the EXPORT command.

Example

For a report file, export all complete Abend-AID reports for the JOBNAME beginning with AACNT to a tape. Delete all entries that have been successfully exported.

```
EXPORT FROMOBJECT=DATABASE,FROMDD=ABNLFROM,TODD=ABNLTO,
       JOBNAME=AACNT*,INC=NO,DELETE=YES
```

IMPORT

The IMPORT command selectively restores previously exported entries from a sequential file to a shared directory or report file.

The “from” file (the sequential file) can be specified using either the FROMDSN and FROMVOL parameters, or by using the FROMDD parameter. The “from” file can also be a file created by the batch DDIO utility CWDDSUTL EXPORT command for a DDIO report file. The “to” file (the shared directory or report file) can be specified using either the TODSN and TOVOL parameters, or by using the TODD parameter.

When an entry is imported to a report file that is attached to a shared directory, the shared directory is also updated. If no other parameters are specified, then all entries are imported.

Note: If the access method is **VSAM**, then you must specify **DISP=SHR** for the TODSN DD statement in your JCL.

The following diagram illustrates the syntax for the IMPORT command.

```
►►─IMPORT──┬─TOOBJECT=SHRDIR──────┬─,──►
           └─TOOBJECT=DATABASE────┘

  ┌─FROMDD=ABNLFROM──────────────┐      ┌─TODD=ABNLTO──────────────┐
►─┼──────────────────────────────┼─,──┼──────────────────────────┼─,──►
  ├─FROMDD=ddname ───────────────┤      ├─TODD=ddname───────────────┤
  └─FROMDSN=dsn,─┬──────────────┘      └─TODSN=dsn,─┬──────────────┘
                 └─FROMVOL=volser┘                  └─TOVOL=volser┘

►─┬──────────────────────┬─,─┬──────────────────┬─,─┬──────────────┬─,──►
  └─ReptNumber=xxxxxxxxxx┘   └─JOBNAME=xxxxxxx*─┘   └─ABEND=xxx*───┘

►─┬──────────────────────────────────────────────────────────────┬─,──►
  ├─DAYSOLD=nnn──────────────────────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,─┬──────────────────────────────────────┘
                          └─EndDATE=mm/dd/yyyy──────────────────┘

►─┬──────────────────────────────────────────────────────────────┬─,──►
  │                                                              │
  └─StartTIME=hh:mm[:ss],─┬─EndTIME=24:00:00──────────────────┬─┘
                          └─EndTIME=hh:mm[:ss]────────────────┘

  ┌─DUPS=ALL──────────┐      ┌─INC=YES──────┐      ┌─NOTFOUND=ERROR──────┐
►─┼───────────────────┼─,──┼──────────────┼─,──┼─────────────────────┼─,──►
  ├─DUPS=LOCKED───────┤      ├─INC=NO───────┤      └─NOTFOUND=IGNORE──────┘
  └─DUPS=UNLOCKED─────┘      └─INC=ONLY─────┘

  ┌─CONTINUE=YES──────┐      ┌─LABEL=1──────┐
►─┼───────────────────┼─,──┼──────────────┼──────────────►◄
  └─CONTINUE=NO───────┘      └─LABEL=nn─────┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5. Refer to Shared Services SLCXCNTL installation library member CXAAIMPO for sample JCL for the IMPORT command.

Example

Restore all entries from a sequential file to a report file.

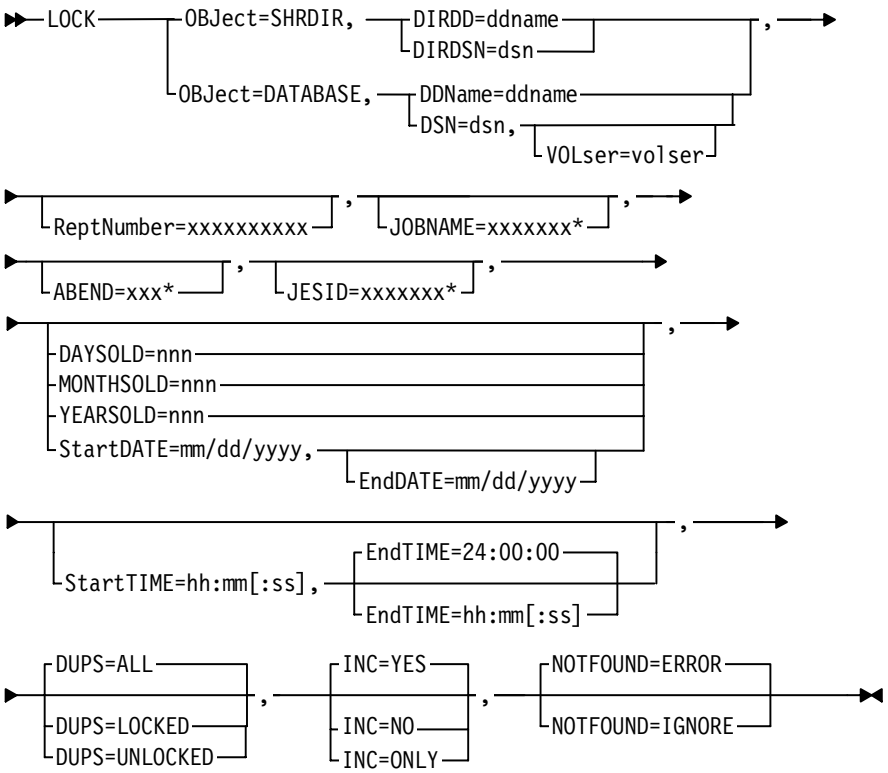
```
IMPORT TOOBJECT=DATABASE,FROMDSN=EFHBMM0.DB01,
       TODSN=COMPWARE.AA.RPT02
```

LOCK

The LOCK command locks selected entries of a specified report file or all report files attached to a specified shared directory.

While an entry is locked with this command (i.e., manually locked), it will not be automatically deleted when the report file becomes full.

The following diagram illustrates the syntax for the LOCK command.



For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

For a report file, lock report number 1105.

```
LOCK OBJECT=DATABASE,DSN=COMPWARE.AA.RPT02,RN=1105
```

MOVE

The MOVE command copies selected entries from one shared directory or report file to another shared directory or report file and deletes them from the “from” file if they were successfully copied. Entries can be moved in any of the following ways:

- from one report file to another report file
- from a report file to a shared directory
- from a shared directory to a report file
- from one shared directory to another shared directory.

Use the FROMDD and TODD parameters to specify the “from” and “to” file ddnames.

When an entry is moved to or from a report file that is attached to a shared directory, the shared directory is also updated. If no other parameters are specified, then all entries are moved.

Note: Only successfully copied entries are deleted from the “from” file.

The following diagram illustrates the syntax for the MOVE command.

```
                                   ┌TODD=ABNLTO──────────────┐
►►─MOVE─┬TOOBJECT=SHRDIR──┬,──┼──────────────────────────┼,──►
        └TOOBJECT=DATABASE┘   ├TODD=ddname──────────────┤
                              └TODSN=dsn,─┬──────────────┘
                                          └TOVOL=volser┘
                                ┌FROMDD=ABNLFROM──────────┐
►──┬FROMOBJECT=SHRDIR──┬,──┼──────────────────────────┼,──►
   └FROMOBJECT=DATABASE┘   ├FROMDD=ddname ──────────────┤
                           └FROMDSN=dsn,─┬──────────────┘
                                         └FROMVOL=volser┘
   ┌RETAINRN=NO─┐
►──┼────────────┼,──┬──────────────────┬,──┬──────────────┬,──►
   └RETAINRN=YES┘   └ReptNumber=xxxxxxxxxx┘   └JOBNAME=xxxxxxx*┘
►──┬──────────────┬,──┬──────────────┬,──┬──────────────┬,──►
   └JESID=xxxxxxx*┘   └ABEND=xxx*────┘   └SORT=sortkey┘
►──┬──────────────────────────────────────────────────┬,──►
   ├DAYSOLD=nnn──────────────────────────────────────┤
   ├MONTHSOLD=nnn────────────────────────────────────┤
   ├YEARSOLD=nnn─────────────────────────────────────┤
   └StartDATE=mm/dd/yyyy,─┬──────────────────────────┘
                          └EndDATE=mm/dd/yyyy────┘
►──┬──────────────────────────────────────────────────┬,──►
   │                        ┌EndTIME=24:00:00────┐   │
   └StartTIME=hh:mm[:ss],──┼────────────────────┼───┘
                            └EndTIME=hh:mm[:ss]──┘
   ┌DUPS=ALL────────┐   ┌INC=YES──┐   ┌CONTINUE=YES─┐
►──┼────────────────┼,──┼─────────┼,──┼─────────────┼,──►
   ├DUPS=LOCKED─────┤   ├INC=NO───┤   └CONTINUE=NO──┘
   └DUPS=UNLOCKED───┘   └INC=ONLY──┘
   ┌NOTFOUND=ERROR──┐
►──┼────────────────┼──►◄
   └NOTFOUND=IGNORE─┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5. Refer to Shared Services SLCXCNTL installation library member CXAAMOVE for sample JCL for the MOVE command.

Examples

Move all entries from one shared directory to another, and do not retain the report numbers.

```
MOVE TOOBJECT=SHRDIR,TODSN=COMPWARE.AA.SHRDIR1,
     FROMOBJECT=SHDIR,FROMDSN=COMPWARE.AA.SHRDIR2,
     RETAINRN=NO
```

Move all unlocked entries from one report file to another report file.

```
MOVE TOOBJECT=DATABASE,
     TODSN=COMPWARE.AA.DB01,
     FROMOBJECT=DATABASE,
     FROMDSN=COMPWARE.AA.DBAA01,
     DUPS=UNLOCKED
```

RESET

The RESET command disconnects a report file from a shared directory when the shared directory was deleted or damaged. The report file can then be attached to another shared directory.

The following diagram illustrates the syntax for the RESET command.

```
►►─RESET─┬─DDName=ddname─┬──────►◄
         └─DSN=dsn───────┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Disconnect a report file from a damaged shared directory by resetting all the flags.

```
RESET DSN=COMPWARE.AA.DB01
```

SCRATCH (SCR)

The SCRATCH command clears all entries from a shared directory for a specified report file that has been deleted or damaged.

The following diagram illustrates the syntax for the SCRATCH command.

```
►►─SCRatch──┬─DIRDD=ddname──┬─,-DSN=dsn──────►◄
            └─DIRDSN=dsn────┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Scratch all entries from a shared directory for a lost report file.

```
SCRATCH DIRDSN=COMPWARE.AA.SHRDIR1,
        DSN=COMPWARE.AA.RPTDB01
```

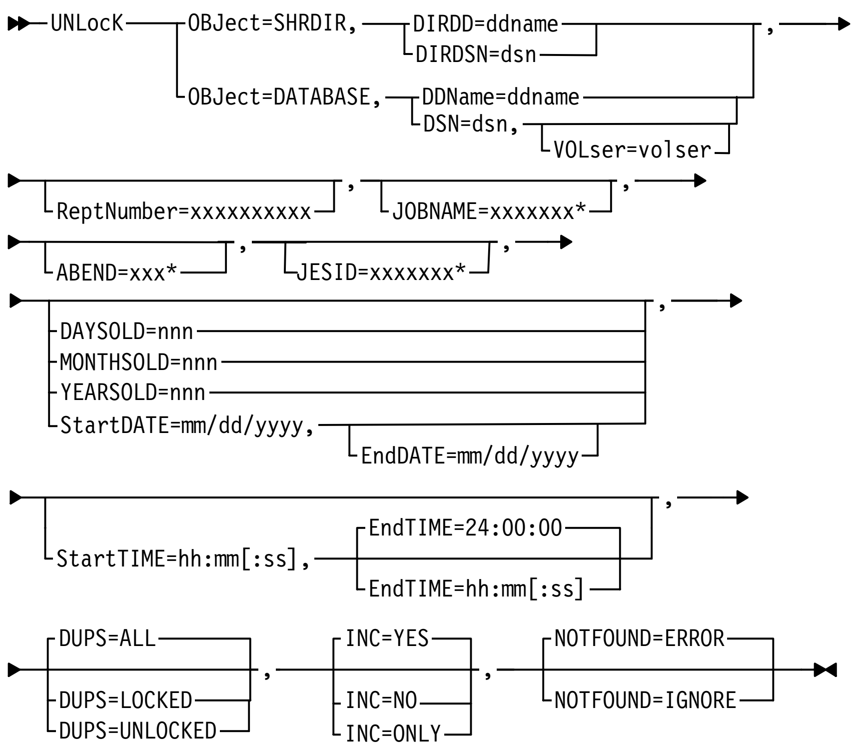
UNLOCK (UNLK)

The UNLOCK command unlocks selected entries of a specified report file or all report files attached to a specified shared directory.

When an entry is unlocked, it can be automatically deleted when the report file becomes full if the report file has been formatted with AUTODELETE=YES or AUTODELETE=DUPS.

At least one selective parameter must be specified.

The following diagram illustrates the syntax for the UNLOCK command.



For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

Unlock entries in report files attached to a shared directory that are older than six months.

```
UNLOCK OBJECT=SHRDIR,
       DIRDSN=COMPWARE.AA.SHRDIR2,MONTHSOLD=6
```

VERIFY (VER)

The VERIFY command examines a selected shared directory to ensure the following:

- All report files attached to the shared directory exist.
- All entries in the shared directory exist in the attached report files.
- All entries in the attached report files are also in the shared directory.

Note: This command creates a report of any inconsistencies but does not perform any maintenance or repairs. See the EXAMINE REPAIR command for automatic repair of any inconsistencies found.

The following diagram illustrates the syntax for the VERIFY command.

```
►►─VERify──┬─DIRDD=ddname─┬──────────────►◄
           └─DIRDSN=dsn───┘
```

For a description of the parameters see “Command Parameters for Shared Directories and Report Files” on page 6-5.

Example

For a shared directory, verify that all attached report files exist and that the entries in the report file match the entries in the shared directory.

```
VERIFY DIRDSN=COMPWARE.ABENDAID.SHRDIR
```

Note: You may use the EXAMINE command to repair any inconsistencies in the Shared Directory that the VERIFY command reports. See “EXAMINE” on page 6-26 for more information about the EXAMINE command.c

Chapter 7.
Batch File Utility CWDDLPUT

This chapter describes the CWDDLPUT batch file utility program used by the language processor for C language, VisualAge PL/I, and Enterprise PL/I processing if using LONGNAME compiler support. These new file types must be managed by CWDDLPUT. The DDIO file utility, CWDDSUTL, may not be used for file maintenance of any of these new file types.

CWDDLPUT allows you to manage a shared directory (SRCEDIR), and its associated Source Listing Databases. The shared directory is a VSAM RRDS file containing a repository of all of the objects necessary to process a compile listing with LONGNAME support contained in a Source Listing Database. The Source Listing Database may be either a VSAM file or a sequential file.

The term entry or member, throughout this chapter, refers to a program source listing in a Source Listing Database.

The terms shared directory or SRCEDIR, throughout this chapter, refer to a repository file containing the names of all Source Listing Databases and the member names of programs processed by the language processor. These member names may be greater than eight characters, in support of C language, VisualAge PL/I, and Enterprise PL/I long external names (LONGNAME).

For information on the CWDDSUTL utility for handling source listing files and Abend-AID report files, refer to **Chapter 4, “Batch File Utility CWDDSUTL”**.

For information on the CWFXSDUT utility for handling Abend-AID for CICS transaction databases and shared directories, refer to **Chapter 5, “Batch File Utility CWFXSDUT”**.

For information on the CWAASDUT utility for handling shared directories and Abend-AID report databases, refer to **Chapter 6, “Batch File Utility CWAASDUT”**.

The term database refers to a specially formatted DDIO source listing file, that is owned and managed by a shared directory (SRCEDIR).

CAUTIONS:

CWDDLPUT is the only utility recommended for use with Compuware Source Listing Shared Directories and Source Listing Databases.

CWDDLPUT must be used to delete Source Listing Databases and/or Source Listing Shared Directories to avoid possible internal corruption.

File-AID/MVS and other VSAM-related vendor products (including vendor packages that compress or enhance buffer management on VSAM files) should NOT be used against Compuware DDIO files. Doing so can corrupt the contents of the database and/or shared directory file.

Summary of CWDDLPUT Utility Commands

The following is a list and brief description of the commands available for the CWDDLPUT batch file utility. Each command is discussed in detail later in this chapter.

ATTACH	Reconnects a detached Source Listing Database to a shared directory.

CHANGE Changes the auto-delete characteristic for a Source Listing Database file or changes the attributes of a DBMODEL previously created.

CONVERT Converts a standard source listing DDIO file to a Source Listing Database and connects it to a shared directory.

COPY Copies entries from one shared directory or database to another shared directory or database.

CREATE Allocates and formats a shared directory or database file. May also be used to format a pre-allocated shared directory or database file. Creates a DBMODEL entry in the shared directory that will be used in conjunction with the new dynamic Source Listing Database create feature (DYNCREATE).

DELETE Scratches a shared directory, database file, DBMODEL entry, or database member entries.

DETACH Disconnects a database file from a shared directory.

DIRECTORY Lists the entries in a shared directory or database file.

DIRS Lists the directory entries in a sequential file that are produced by the EXPORT command.

DIRX Reports on the characteristics of a shared directory and all Source Listing Databases owned by the shared directory. Reports on the characteristics of the entries in a Source Listing Database.

ESTIMATE Calculates the number of blocks to allocate for a shared directory when the allocation is performed outside of CSS.

EXPAND Resizes an existing shared directory's space allocation and/or CISIZE.

EXPORT Copies entries from a shared directory or Source Listing Database to a sequential file, from which they can later be restored into the same or another shared directory or Source Listing Database using the IMPORT command.

IMPORT Restores previously exported entries from a sequential dataset.

LOCK Locks the entries of a Source Listing Database, or the entries of all Source Listing Databases attached to a specific shared directory, so they cannot be automatically deleted (see AUTODELETE) when the file becomes full.

MOVE Copies entries from one Shared Directory or Source Listing Database to another Shared Directory or Source Listing Database and deletes them from the "from" file only if they were successfully copied.

PRINT Prints a single source member from a Source Listing Database or Shared Directory or prints a DBMODEL's specifications.

RESET Disconnects a Source Listing Database from a shared directory when the shared directory is lost or damaged.

SCRATCH Removes the entries in a shared directory for a Source Listing Database that has been lost or damaged.

TEST Allows a test create and format of a Source Listing Database as specified in the DBMODEL used for the dynamic database create feature (DYNCREATE). The file will not be attached to the shared directory.

UNCONVERT Detaches the Source Listing Database from its shared directory and then converts it back to a standard DDIO source listing file format. Long-name members will still be in the converted DDIO source listing file, but will not correctly identify themselves and will not be usable.

UNLOCK Unlocks selected entries of a specified Source Listing Database, or all Source Listing Databases attached to a specific shared directory, so that they can be automatically deleted when the Source Listing Database file becomes full.

VERIFY Ensures that the information contained in a shared directory matches the information contained in its Source Listing Databases. This command produces a report that lists any inconsistencies.

Usage Considerations

This section discusses special considerations for using the CWDDLPUT utility.

Renaming Shared Directories and Source Listing Databases

To rename a Source Listing Database, you must first detach it from the shared directory and then rename it. You must then re-attach it to the shared directory in order to retain all the correct references. The newly renamed databases must be connected to the shared directory before it can be used.

To rename a shared directory, you must first detach all the Source Listing Databases from the old shared directory. Then, re-attach the Source Listing Databases to the renamed shared directory.

Dynamic Creation of Source Listing Databases

This new feature allows dynamic creation of a new Source Listing Database, during creation of a new source listing member, if all attached files in the shared directory are not selectable (full, or not enough space for that member) and you specified CWPDDIO as a shared directory. You must also have previously set up a valid DBMODEL with DYNCREATE=YES in that shared directory as described here using CREATE. The dynamic create function is NOT invoked during the CWDDLPUT utility functions ATTACH, COPY, MOVE, IMPORT, or CONVERT.

DD Statements Used with CWDDLPUT

Following is a description of each DD statement required for CWDDLPUT. Table 7-1 on page 7-4 lists the required and optional DD statements used with CWDDLPUT batch commands.

ABNLPARM

The ABNLPARM DD statement specifies the control dataset. Normally, the control dataset resides in the input stream; however, it can reside on a system input device or it can be defined as a member in a partitioned dataset. You must specify the ABNLPARM DD statement each time you run the batch file utility. The ddname ABNLPARM cannot be altered.

ABNLREPT

The ABNLREPT DD statement specifies a sequential dataset for messages and reports produced by CWDDLPUT. You must specify the ABNLREPT DD statement each time you run the batch file utility. The ddname ABNLREPT cannot be altered.

ABNLFROM

The ABNLFROM DD statement specifies the input Source Listing Database for the COPY, EXPORT, IMPORT, and MOVE commands. The default ddname ABNLFROM can be changed using the FROMDD parameter.

ABNLTO

The ABNLTO DD statement specifies the output Source Listing Database for the COPY, EXPORT, IMPORT, and MOVE commands. The default ddname ABNLTO can be changed using the TODD parameter.

anyname1

The anyname1 DD statement specifies an optional sequential dataset for a printed member for the PRINT command. The actual ddname is defined by the OUTDD parameter of the PRINT command. If this parameter is not defined, the member will be written to the file specified by the ABNLREPT DD statement.

The dataset specified by the anyname1 DD statement must use RECFM=FBM and LRECL=133. The BLKSIZE must be a multiple of the LRECL value. If the dataset is a new dataset (DISP=NEW) and the DCB parameters are not specified, the default BLKSIZE value is 133.

Table 7-1. DD Statements Used with CWDDLPUT Batch Commands

	ABNLPARM	ABNLREPT	ABNLFROM	ABNLTO	anyname1
ATTACH	X	X			
CHANGE	X	X			
CONVERT	X	X			
COPY	X	X	O	O	
CREATE	X	X			
DETACH	X	X			
DELETE	X	X			
DIR	X	X			
DIRS	X	X			
DIRX	X	X			
ESTIMATE	X	X			
EXPAND	X	X			
EXPORT	X	X	O	O	
IMPORT	X	X	O	O	
LOCK	X	X			
MOVE	X	X	O	O	
PRINT	X	X			O
RESET	X	X			
SCRATCH	X	X			
TEST	X	X			
UNCONVERT	X	X			
UNLOCK	X	X			
VERIFY	X	X			
X – Required; O –Optional					

Usage Restrictions

1. CWDDLPUT is the only utility recommended for use with Compuware Source Listing Shared Directories and Source Listing Databases.
2. CWDDLPUT must be used to delete Source Listing Shared Directories and/or Source Listing Databases to avoid possible internal corruption.
3. File-AID/MVS and other VSAM-related vendor products (including vendor packages that compress or enhance buffer management on VSAM files) should NOT be used against Compuware DDIO files. Doing so can corrupt the contents of the database and/or shared directory file.

Command Conventions

Use the following conventions when specifying commands:

- Enter commands and their parameters as control cards in the dataset defined by the ABNLPARM DD statement.
- Commands can start in any column.
- Separate the command from its parameters with one or more spaces.
- Separate parameters with commas without intervening spaces.
- When typing parameter names, type either the whole parameter name, the shortcut in parentheses in the parameters list (see "Command Parameters for Shared Directories and Source Listing Databases" on page 7-6), or the uppercase letters shown in the command syntax diagrams. Typing a shortened version of the parameter WILL NOT work. For example, typing AUTOD for the AUTODELETE parameter is not valid — you must type AUTODELETE or AD. The same rules apply to typing command names.
- Control cards can be continued on successive lines. Each line of a multiple-line control card statement must contain at least one parameter. To continue on a new line, enter a comma followed by a space immediately after the last parameter on the line. EXCEPTION: to accommodate long member names, the PROGRAM (P) utility parameter may be continued by coding the name through column 80 and continuing in column 1 of the next line or lines. The member name may be up to 256 characters (mixed case), and may include the asterisk (*) wild card character at the end.
- To include a comment on a line, enter at least one blank space after the last parameter on the line followed by the comment text.
- Any line containing an asterisk (*) followed by a space in the first two columns is treated as a comment line.
- The CWDDLPUT utility stops when it reaches an end-of-file (EOF) or null input control card.
- The asterisk (*) is the generic or wildcard character indicator, and it can be used with the following parameter:
  - PROGRAM

  The wildcard character can be used only at the end of the parameter being specified.
- Parameter strings with blanks must have apostrophes around them.

Command Parameters for Shared Directories and Source Listing Databases

This section describes the parameters accepted by the commands that manage existing shared directories, Source Listing Databases, and DBMODEL (used for dynamic creation feature). Not all parameters are available for all commands.

AM:

Specifies the access method for the file being created. This parameter is only used when allocating or reallocating a file. Allowable values are:

VSAM: The access method is VSAM.

BSAM: The access method is sequential.

Note: A shared directory must be created as a VSAM file.

AUTODELETE (AD):

Determines the action taken when an attempt is made to write a member to a Source Listing Database file that is full. The AUTODELETE parameter specifies whether the oldest member in the Source Listing Database file is automatically deleted when the file or directory becomes full. A Source Listing Database file is considered full when all of the groups (EXTENTs) in the file have been allocated or the number of members specified by REPTCOUNT has been reached.

The autodelete process is based on **manual** and **automatic** locks. Manual locks are created online using the **L** line command from any directory screen or with the batch utility LOCK command. Automatic locks are created whenever a member is added to a Source Listing Database file formatted using the AUTODELETE=DUPS or AUTODELETE=STAGED options. The most recent member on that database is automatically locked.

Members that have been manually locked retain their locks during the autodeletion process. They may be unlocked by using the **U** line command online or with the UNLOCK batch command. A DELETE command in batch, or the **D** line command online, will unlock a member prior to executing the delete.

When a source shared directory is specified, the autodelete attributes are reflected at the database level. For language processor *source-member-create*, all attached databases participate in the selection process based on their free space amounts. After selection, that database's autodelete setting determines any autodeletion and automatic locking.

For source listings, the oldest member is the program with the earliest compile date and time. The following are valid values.

YES: (Default): Allows new members to be written to a full DDIO file by deleting the oldest unlocked members. This method is commonly referred to as **wrap-around** mode, and is typically used for source listing files where members are added as needed. AUTODELETE=YES does not automatically lock members.

A file full condition will occur when AUTODELETE=YES and all members are manually locked or in use. You can resolve a file full condition by manually deleting members from the file, manually unlocking members, or using the DBMODEL feature.

Notes:

1. If you want to keep a specific member for a longer period, you can lock it, and it will be bypassed when autodelete processing selects the next report member to be automatically deleted.
2. The DELETE DUPS parameter should only be used against AUTODELETE=DUPS or AUTODELETE=STAGED files. Using DELETE DUPS=UNLOCKED against an AUTODELETE=YES file will delete all unlocked members ('ALL' members).

DUPS: This option ensures that one entry of each name remains in a Source Listing Database. When a new member is added to a Source Listing Database file, all automatically locked members with the same name are unlocked except the member with the most recent compile date and time. Members with the same program name that have been manually locked using the **L** line command or the LOCK batch command remain locked and will not be deleted. If the Source Listing Database file is full, the oldest unlocked member not in use, regardless of name, is chosen for autodeletion. When "OBJECT=SRCEDIR" is specified for member create, each of its attached databases will participate in the selection process, but only the selected database (if it is DUPS) participates in this automatic unlock-lock process; the most current duplicate member for the selected database will be locked if it is DUPS.

A file full condition will occur when all members are manually or automatically locked or in use. You can resolve a file full condition by manually deleting members from the file, manually unlocking members, or by using the DBMODEL feature.

STAGED: This option is similar to AUTODELETE=DUPS, and is valid for source listings only.

When a new member is added to a source listing database, all automatically locked members with the same name are unlocked except the member with the most recent compile date and time. Members with the same name that have been manually locked remain locked and will not be deleted. If the source listing database is full, the oldest unlocked member not in use is chosen for autodeletion. When "OBJECT=SRCEDIR" is specified for member create, each of its attached databases will participate in the selection process, but only the selected database (if it is STAGED) participates in this automatic unlock-lock process. The most current duplicate member for the selected database will be locked (if it is STAGED).

However, if all members are locked or in use, then the oldest automatically locked member not in use is deleted.

CAUTION:

If you need to retain the most current version of each source listing member, you should format your listing file with AUTODELETE=DUPS or AUTODELETE=NO.

A file full condition will occur when all members are manually locked or in use. You can resolve a file full condition by manually deleting members from the file, manually unlocking members, or by using the DBMODEL feature.

NO: Prevents members from being written to a Source Listing Database file when it becomes full. To add members to a full file, one or more members of the file must be manually deleted using the DELETE command.

If the Source Listing Database file holds production source listings, you may want to use AUTODELETE=NO so that the production listings are not automatically deleted before a new member is successfully added. A Source Listing Database file that holds listings from development could be formatted with AUTODELETE=DUPS or AUTODELETE=STAGED. With the AD=DUPS or AD=STAGED option, new members may be added to the full DDIO file if there are any members eligible to be automatically deleted.

A file full condition will occur when the file is full. You can resolve a file full condition by manually deleting members from the file or by using the DBMODEL feature.

Note: A DIR/DIRX member list of the shared directory will reflect the mixture of database-level locking attributes.

DDIO File Full Condition

A **file full condition** exists when new members cannot be added to a full DDIO file, in which case an error is returned. A DDIO file full condition will occur with the following AUTODELETE options:

YES	All members are manually or automatically locked or in use.
DUPS	All members are manually or automatically locked or in use.
STAGED	All members are manually locked or in use.
NO	The file is full.

Resolving a File Full Condition

You can perform the following steps to resolve a file full condition:

- Manually delete members from the file.
- Manually unlock members when AUTODELETE=YES, AUTODELETE=DUPS, or AUTODELETE=STAGED.
- Allocate and format a larger file, and use the MOVE or COPY commands to transfer the contents of the full file to the new file.
- Use the EXPORT command with the DELETE option to archive all or some members to a sequential file (tape or disk).

  Using EXPORT, space can be freed up without losing members, since members can be imported back into the file as needed. This method also avoids the overhead created by formatting new files.
- Set up your shared directory to perform dynamic database creation. You must first create a DBMODEL in your shared directory, which defines the parameters that will be used to create and format a new Source Listing Database dynamically. (See the CREATE command, “CREATE (CRE)” on page 7-22.)

  The DYNCREATE parameter must also be set to YES. The CWPDDIO DD statement must point to your shared directory (NOT to a specific Source Listing Database). When the file-full condition is encountered, a new Source Listing Database will dynamically be created and formatted and the source member created in that database. An informational message is generated to let you know this has occurred.

BLKSIZE (BLK):

If you specified AM=BSAM, use this parameter to specify a blocksize for the sequential Source Listing Database. The default is 4096 (or 4 KB) and the maximum BLKSIZE is 32767. The BLKSIZE should be between 2048 (or 2 KB) and the track size. If you pre-allocated your database using AM=BSAM, you must still specify BLKSIZE, or it will default to 4096 during format. The blocksize specified at original create is ignored.

CISIZE:

If you specified AM=VSAM, use this parameter to specify the control interval size.

If neither the CISIZE nor the BLKSIZE parameter is specified, then the default CISIZE is 4096, and the default blocksize is 4089.

If the CISIZE parameter is not specified, but the BLKSIZE parameter is specified, then the value for CISIZE is determined by adding 7 to the value of the BLKSIZE parameter.

If the BLKSIZE parameter is not specified, but the CISIZE parameter is specified, then the blocksize is determined by subtracting 7 from the value of the CISIZE parameter.

If both the CISIZE and the BLKSIZE parameters are specified, then the specified BLKSIZE parameter value is ignored. In this case, the blocksize is determined by subtracting 7 from the value of the CISIZE parameter.

CONTINUE:

Specifies whether the command will continue if recoverable errors are detected. A value of YES (default) specifies that the command will continue, if possible, when a recoverable error occurs. A value of NO specifies that the command will be stopped when an error occurs. This parameter is valid for the EXPORT, IMPORT, COPY, and MOVE commands.

CYLINDERS (CYL):

To use cylinders as the unit of measure for the size of the file, use this parameter to specify the number of cylinders. This parameter is valid for both VSAM and sequential files.

DATABASEDISP or DBDISP:

When a shared directory is being deleted, this parameter determines the disposition of the Source Listing Databases attached to the shared directory. Allowable values are:

DETACH: (Default) Detaches all attached Source Listing Databases from the shared directory prior to deleting the shared directory. This choice preserves the Source Listing Databases and allows you to attach them to another shared directory.

DELETE: Does not detach the Source Listing Databases from the shared directory. This choice deletes both the shared directory and the attached Source Listing Databases.

DATACLAS:

Specifies the eight-character data class name for the Storage Management Subsystem (SMS) operating environment. This parameter is valid only for new shared directories or Source Listing Databases, or when the REALLOCATE parameter is set to YES.

DAYSOLD:

Allows you to select entries in a shared directory or Source Listing Database that are older than nnn days. For example, using the DELETE command for a Source Listing Database with DAYSOLD=30 deletes all source listing entries that are more than 30 days old.

DDNAME (DDN):

If you specified OBJECT=DATABASE, use this parameter for the ddname of the JCL statement that specifies the dataset name of the Source Listing Database.

DELETE:

Specifies whether successfully exported entries will be deleted from the "from" Source Listing Database. A value of YES specifies that the entries will be deleted. A value of NO specifies that the entries will not be deleted. The default value is NO.

DIRDD:

If you specified OBJECT=SRCEDIR, this parameter indicates the ddname of the shared directory. If you specified OBJECT=DATABASE, this parameter indicates the ddname of the shared directory the Source Listing Database file is attached to.

DIRDSN:

If you specified OBJECT=SRCEDIR and will not be using a JCL statement for the dataset name, use this parameter to specify the dataset name of the shared directory. This parameter contains the actual dataset name.

If you specified OBJECT=DATABASE and will not be using a JCL statement for the dataset name of the shared directory, use this parameter to specify the dataset name of the shared directory to which the Source Listing Database will be attached.

Note: You can use either the DIRDD or the DIRDSN parameter, depending on whether you are using a JCL statement for the dataset name of the shared directory.

DIRENTS:

If you specified OBJECT=SRCEDIR, and you are unable to determine the number of blocks you will need for the Shared Directory, use this parameter to allocate an appropriate number of blocks. This parameter specifies the total number of directory entries expected in the Shared Directory. The total number of directory entries consists of the total number of source listings for each Source Listing Database attached to the Shared Directory. When combined with a blocksize, the DIRENTS parameter is used by CWDDLPUT to calculate a reasonable number of blocks for allocation. The blocksize can be specified directly in the BLKSIZE parameter, or indirectly through the CISIZE or MODEL parameter. If you will be using an external method to allocate blocks (e.g., IDCAMS or Compuware's File-AID), you can use the ESTIMATE command to determine the number of blocks you will need. See "ESTIMATE" on page 7-33 for more information on the ESTIMATE command.

DIRVOL:

If you specified OBJECT=SRCEDIR and will not be using a JCL statement for the volume serial of the shared directory, you can use this parameter to specify the volume serial number.

DSN:

If you specified OBJECT=DATABASE and are **not** using a JCL statement for the dataset name, use this parameter to specify the dataset name of the Source Listing Database.

If you specified OBJECT=DBMODEL, the DSN specification "dbmodelname" is stored and used later to dynamically build the actual DSN of the Source Listing Database that is created during the "dynamic database create function" (DYNCREATE). When the DSN is specified for either CREATE or CHANGE of a DBMODEL, certain "tokens" are allowed for specifying DSN qualifiers along with (or in place of) literals. The tokens are prefixed with an "&" and must be separated by a ".". If the combined length of literals, generated qualifiers, and separators is greater than 44, an error will

occur when trying to build the name. The TEST function is available to "test" the actual DSN creation. The allowable tokens are:

&JOB insert job name as qualifier

&STEP insert job step as qualifier

&USER insert user/owner ID (if the user/owner ID is unavailable the token will be treated as NULL)

&DATE insert current date in Julian format preceded by the letter "D" (YYYYDDD)

&TIME insert current time preceded by the letter "T" (HHMMSS format)

&SEQ insert a sequentially increasing number preceded by the letter "S" (NNNNNNN)

Example:

To build a DSN with high-level qualifier of job name, literal "test", date, time specify:

```
DSN=&JOB..TEST.&DATE..&TIME
```

DUPS:

Specifies whether all entries, only locked entries, or only unlocked entries are selected. The available values are: ALL, LOCKED, or UNLOCKED. The default value is ALL.

DYNCREATE:

If set to YES, specifies that dynamic database creation will be invoked if the attached databases do not have enough space to create a new source listing member. The default value is NO. This is used only with OBJECT=DBMODEL.

ENDDATE (EDATE):

Specifies the last compile date of the source listings. This parameter can be specified as either MM/DD/YY or MM/DD/YYYY. Acceptable four-digit years are 1970 through 2069. If ENDDATE is specified, then STARTDATE must also be specified. The highest valid ENDDATE is 12/31/2069.

ENDTIME (ETIME):

Specifies the last compile time of the source listings. This parameter must be specified as HH:MM[:SS] or HH.MM[.SS]. If seconds are omitted, the default is 59.

This parameter must be preceded by the STARTTIME parameter. If the STARTTIME parameter is specified and the ENDTIME parameter is omitted, then ENDTIME defaults to 24:00:00.

When using STARTTIME and ENDTIME in conjunction with STARTDATE and ENDDATE, entries are first selected by date, then by time.

EXTENTS:

Specifies the maximum number of extents to be allocated for each Source Listing Database entry. If the EXTENTS parameter is not specified, the default value is 64. The maximum value is 460.

FROMDD:

Specifies the ddname of the Source Listing Database or shared directory from which the entries are copied, exported, or moved. It also specifies the ddname of the sequential file from which previously exported entries are imported.

FROMDSN:

Specifies the DSN of the Source Listing Database or shared directory from which the entries are copied, exported, or moved. It also specifies the DSN of the sequential file from which previously exported entries are imported.

FROMOBJECT:

Specifies the type of file from which the entries are copied, exported, or moved. Allowable values are:

SRCEDIR: The "from" file is a shared directory.

DATABASE: The "from" file is a Source Listing Database.

FROMVOL:

Specifies the VOLSER of the Source Listing Database or shared directory from which the entries are copied, exported, or moved. It also specifies the VOLSER of the sequential file from which previously exported entries are imported.

GROUPCOUNT (GC):

Specifies the number of blocks allocated to a Source Listing Database entry each time additional space is needed. The minimum value is 1, and the maximum value is 2048. The default value is 4.

INC:

Specifies whether to include incomplete entries in the selection. Allowable values are:

YES: (Default) Select both complete and incomplete entries.

NO: Select only complete entries.

ONLY: Select only incomplete entries.

LABEL:

Specifies the file number for the sequential dataset on the tape for the IMPORT, EXPORT, and DIRS commands. (VOL=SER=, and other parameters should be specified on the corresponding DD statement). The default value is 1. The LABEL parameter allows you to reference multiple export files using a single ddname. This provides a means by which you can create multiple export files on a single tape. For example, to include a Source Listing Database on a tape that already contains the related source code and compiler output as files 1 and 2, you can specify LABEL=3 in the EXPORT command for that Source Listing Database. This will append the Source Listing Database on the tape as file 3. To perform a DIRS or IMPORT on that file at a later time, include LABEL=3 with the command. The LABEL parameter is valid for IMPORT, EXPORT, and DIRS. You can include the LABEL parameter in your JCL.

Note: If you include the LABEL parameter in your JCL, but have also specified a LABEL parameter with the IMPORT or EXPORT batch command, the LABEL parameter in the JCL will be overridden.

For compatibility with the IBM Tape Management System, the label number ".Lnnn" is appended to the tape DSNAME as the last qualifier. If the combined length of the specified DSNAME and the appended label number exceeds the maximum length for DSNAME, an error message is issued.

MGMTCLAS:

Specifies the eight-character management class name for the Storage Management Subsystem (SMS) operating environment. This parameter is only valid for new shared directories or Source Listing Databases, or when the REALLOCATE parameter is set to YES.

MODEL:

Use this parameter to specify the DSN of an existing shared directory or Source Listing Database that can be used as a "template" for creating a new shared directory or Source Listing Database with the same VSAM characteristics. When the MODEL parameter is used, you can override any of the VSAM or BSAM parameters by including the appropriate parameter with the CREATE or EXPAND command. For example, to use a particular shared directory as a template for creating a new shared directory, but with a different CISIZE, include the CISIZE parameter in the CREATE command.

MONTHSOLD:

Allows you to select entries in a shared directory or a Source Listing Database that are older than nnn months. For example, using the DELETE command for a Source Listing Database with MONTHSOLD=6 deletes all source listing entries older than six months.

NOTFOUND:

Specifies whether to request a return code 00, instead of the default code 44, when an entry is not found. The valid values are:

ERROR: (Default) Allows the default code 44.

IGNORE: Returns a return code 00.

OBJECT:

Specifies the type of object being changed, deleted, locked, unlocked, created, reallocated, or formatted. This parameter also specifies the shared directory for the EXPAND command, and the shared directory or the Source Listing Database for the DIRX command. Allowable values are:

SRCEDIR: Shared directory for Source Listing Databases.

DATABASE: Source Listing Database.

ENTRY: Member in the file. This option is used only with the DELETE command when removing an entry from a Source Listing Database and the shared directory to which it is attached.

DBMODEL: Model database specifications for the dynamic database creation feature (DYNCREATE).

OUTDD (O):

Specifies the ddname for the alternative output file for the PRINT member command. If the OUTDD parameter is not specified, the output of the PRINT member command is written to the file specified by the ABNLREPT DD statement, along with all CWDDLPUT messages and reports.

See **anyname1** on page 7-4 for more information on this ddname and the PRINT command.

PROGRAM (P):

Specifies source listings by program name. Wildcard characters are valid, and long names are supported. Long member names may be up to 256 mixed case characters, and the name may be continued by coding the name through column 80 and continuing it on the next line or lines beginning in column 1.

Example:

```
DELETE OBJECT=ENTRY,
DIRDSN=CW.SRCEDIR,P=A_very_long_member_name_from_a_C_language_func
tion_name
```

REALLOCATE (REALLO):

If the shared directory or Source Listing Database you specified already exists, use this parameter to reallocate the file. Allowable values are:

NO: (Default) For sequential files, the file will be reformatted if possible. For VSAM files, an error message is returned and the file is left unchanged.

YES: For both VSAM and sequential files, the file will be deleted and then reallocated.

RECORDS (RECS):

If you specified AM=VSAM and want to use records as the unit of measure for the size of the file, use this parameter to specify the number of records. This parameter is valid for VSAM files only.

REPTCOUNT (RC):

Specifies the maximum number of entries in the Source Listing Database, provided there is enough space. The minimum value is 1, and the maximum number value is 16380. The default is 100.

REPTNUMBER (RN):

Specifies an entry's number. If this parameter is specified, all other selective parameters are ignored. Maximum length is ten digits.

RETAINRN:

Specifies whether the entry's report number will be retained during a move or copy. Allowable values are:

NO: (Default) Does not retain the number.

YES: Retains the number. In this case, if the number to be retained is already being used in the “to” file, then the next highest available number will be used. This value should only be used when moving or copying to a shared directory.

SECONDARY:

This parameter can only be used when formatting pre-allocated shared directories or Source Listing Databases. It allows shared directories or VSAM Source Listing Databases to be allocated with secondary extents. Allocating secondary extents allows a shared directory or VSAM Source Listing Database to be allocated on multiple volumes. The default is NO.

Note: This parameter is only valid for shared directories or VSAM source listing report files, and the VSAM cluster must be pre-allocated.

SORT:

Specifies the order in which entries are selected for the execution of a command. By default, entries are selected in descending date/time order. Entries are first ordered by date, from newest (most recent) to oldest. Entries with the same date are then ordered by time (from latest to earliest). Valid sort keys, their abbreviations, and the sorting sequence they cause are specified in Table 7-2.

Table 7-2. Source Listing Database Sort Keys for XPEDITER/TSO

Sort Key	Abbreviation	Primary Sort Field	Sequence	Secondary Sort Field	Sequence	Tertiary Sort Field	Sequence
PROGRAM	PGM	PROGRAM	Ascending	DATE/TIME	Descending	LIST-NUMBER	Descending

Table 7-2. Source Listing Database Sort Keys for XPEDITER/TSO

Sort Key	Abbreviation	Primary Sort Field	Sequence	Secondary Sort Field	Sequence	Tertiary Sort Field	Sequence
NUMBER	NUM	LIST-NUMBER	Descending				
DATE		DATE/TIME	Descending	LIST-NUMBER	Descending		
RC (return code)		RC	Ascending	DATE/TIME	Descending	LIST-NUMBER	Descending
LANGUAGE	LANG	LANGUAGE	Ascending	DATE/TIME	Descending	LIST-NUMBER	Descending
SIZE		SIZE	Descending	DATE/TIME	Descending	LIST-NUMBER	Descending

STARTDATE (SDATE):

Specifies the compile date of the source listing. The STARTDATE parameter is used with the ENDDATE parameter to specify a date range. The STARTDATE parameter can be specified as either MM/DD/YY or MM/DD/YYYY. Acceptable four-digit years are 1970 through 2069. When this parameter is used alone, all entries for the specified date are selected. The earliest valid start date is 01/01/1970.

STARTTIME (STIME):

Specifies the beginning compile time of the source listings. This parameter must be specified as HH:MM[:SS] or HH.MM[.SS]. If seconds are omitted, the default is 00. This parameter must precede the ENDTIME parameter (if ENDTIME is specified).

If the STARTTIME parameter is specified and the ENDTIME parameter is omitted, then ENDTIME defaults to 24:00:00.

When using STARTTIME and ENDTIME in conjunction with STARTDATE and ENDDATE, entries are first selected by date, then by time.

STORCLAS:

Specifies the eight-character storage class name for the Storage Management Subsystem (SMS) operating environment. This parameter is only valid for new shared directories or Source Listing Databases, or when the REALLOCATE parameter is set to YES.

TODD:

Specifies the ddname of the Source Listing Database or shared directory to which the entries are copied, imported, or moved. It also specifies the ddname of the sequential file to which entries are exported.

TODSN:

Specifies the DSN of the Source Listing Database or shared directory to which the entries are copied, imported, or moved. It also specifies the DSN of the sequential file to which entries are exported.

TOOBJECT:

Specifies the type of file to which the entries are copied, imported, or moved. Allowable values are:

SRCEDIR: The “to” file is a shared directory.

DATABASE: The “to” file is a Source Listing Database.

TOVOL:

Specifies the VOLSER of the Source Listing Database to which the entries are copied, imported, or moved. It also specifies the VOLSER of the sequential file or shared directory to which previously exported entries are exported.

TRACKS (TRKS):

To use tracks as the unit of measure for the size of the file, use this parameter to specify the number of tracks. This parameter is valid for both VSAM and sequential files.

TYPE:

When creating a Source Listing Database, use this parameter to specify the type of entries the file can contain. For a Source Listing Database, code TYPE=SOURCE.

UNIT:

If you specified AM=BSAM, use this parameter to specify a device or device type. The default is SYSDA.

VOLSER (VOL):

If you specified OBJECT=DATABASE, use this parameter to specify the volume serial number of the Source Listing Database when you are using the DSN parameter.

VSAMCAT:

If you specified AM=VSAM, use this parameter to specify the VSAM catalog in which the file will be defined. This is an optional parameter.

YEARSOLD:

Allows you to select entries in a shared directory or Source Listing Database that are older than nnn years. For example, using the DELETE command for a Source Listing Database with YEARSOLD=1 deletes all source listing entries that are older than 1 year.

Command Descriptions and Syntax

This section provides a detailed description of the commands available for existing Source Listing Databases and shared directories including valid command abbreviations and the parameters allowed for each command. With the exception of specifying REPTNUMBER, you can enter more than one selection parameter with a command.

ATTACH (ATT)

The ATTACH command connects a Source Listing Database to a shared directory and populates the shared directory with Source Listing Database entries. A Source Listing Database must be attached to a shared directory before it can be used.

The following diagram illustrates the syntax for the ATTACH command.

```
►►─ATTach──┬─DIRDD=ddname─┬─,──┬─DDName=ddname──┬──►◄
           └─DIRDSN=dsn───┘    └─DSN=dsn────────┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Attach a Source Listing Database to a shared directory.

```
ATTACH DSN=COMPWARE.LP.SOURCE.LISTING.LPDB01,
       DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR
```

Note: If an error occurs during the ATTACH process, when adding members into the shared directory, the database will set “attach in progress” (will show on a DIRX), and the shared directory MAY contain entries for only some (but not all) of the database members. If the problem cannot be solved by another attempt to ATTACH, then perform the following to eliminate possible integrity problems:

Run DETACH on the database to clear the ATTACH setting and remove entries added during the ATTACH process.

CHANGE (CHG)

The CHANGE command changes the AUTODELETE attribute of a Source Listing Database, or changes the attributes of a previously-defined DBMODEL in the specified shared directory.

Change Command – Database to AUTODELETE Attributes

The following diagram illustrates the syntax for the CHANGE command, AUTODELETE attribute.

```
▶▶─CHanGe──OBJect=DATABASE,──┬─DDName=ddname──────────────┬─,──▶
                             └─DSN=dsn,──┬──────────────┬─┘
                                         └─VOLser=volser─┘
   ┌─AutoDelete=YES──┐
▶──┼─────────────────┼──────────────────────────────────▶◀
   ├─AutoDelete=DUPS─┤
   ├─AutoDelete=NO───┤
   └─AutoDelete=STAGED┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Change the AUTODELETE attribute on a Source Listing Database from AUTODELETE=NO to AUTODELETE=DUPS.

```
CHANGE OBJECT=DATABASE,DSN=COMPWARE.SOURCE.LISTING.LPDB01,AD=DUPS
```

CHANGE Command – DBMODEL Attributes

This CHANGE syntax diagram illustrates the syntax for allocating and formatting a new Source Listing Database.

```
►►─CHanGe─OBJect=DBMODEL,─┬─DIRDD=ddname─┬─(1),─DSN=dbmodelnam(2),─►
                          └─DIRDSN=dsn───┘

►┬─AM=VSAM,─┬──────────────┬─,─┬──────────────────┬─,─┬──────────────┬─,─►
│           ├─RECords=nnnn─┤   │ ┌─CISIZE=4096─┐   │   └─VSAMCAT=ddname─┘
│           ├─TRacKS=nnnn──┤   ├─┼─────────────┼───┤
│           ├─CYLinders=nnnn┤   │ └─CISIZE=nnnn─┘   │
│           └─DIRENTS=nnnn─┘   │ ┌─BLKsize=4089─┐   │
│                              └─┼──────────────┼───┘
│                                └─BLKsize=nnnn─┘
│                                ┌─UNIT=SYSDA───┐   ┌─BLKsize=4096─┐
└─AM=BSAM,─┬──────────────┬─,─┼──────────────┼─,─┼──────────────┼──────┘
           ├─TRacKS=nnnn──┤   └─UNIT=xxxxxxxx─┘   └─BLKsize=nnnn─┘
           └─CYLinders=nnnn┘

►─┬──────────┬─,─┬──────────────────┬─,─┬──────────────────┬─,─►
  └─MODEL=dsn─┘   └─MGMTCLAS=xxxxxxxx─┘   └─STORCLAS=xxxxxxxx─┘

                                ┌─ReptCount=100─┐
►─┬──────────────────┬─,─┼───────────────┼─,─────TYPE=SOURCE,─►
  └─DATACLAS=xxxxxxxx─┘   └─ReptCount=nnnn─┘

  ┌─GroupCount=4─┐   ┌─AutoDelete=YES───┐   ┌─EXTENTS=64─┐   ┌─DYNCREATE=NO─┐
►─┼──────────────┼─,─┼──────────────────┼─,─┼────────────┼─,─┼──────────────┼─►◄
  └─GroupCount=nn─┘   │AutoDelete=DUPS───┤   └─EXTENTS=nnn─┘   └─DYNCREATE=YES─┘
                      ├─AutoDelete=NO────┤
                      └─AutoDelete=STAGED─┘
```

Notes:

1. Please note that the DIRDD parameter (or DIRDSN) specifies the name of the shared directory to which you are attaching the Source Listing Database.
2. Please note that the “dbmodelname” used here specifies the name to be created for the new Source Listing Database. It consists of literals and/or symbolic tokens that will generate the final DSN name at creation time. Remember, the total of generated characters plus separator periods must not exceed 44. See the DSN parameter description for more detail.

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Change one or more of the DBMODEL attributes for creating a new Source Listing Database in the specified Source Listing Shared Directory, including turning the DYNCREATE feature on or off.

Change DSN name to be created with date and time included, use VSAM, and set for DUPS:

```
CHANGE OBJECT=DBMODEL,DSN=COMPWARE.LP.SOURCE..&DATE..&TIME,
       AD=DUPS,AM=VSAM,DIRDSN=COMPWARE.LP.SRCEDIR
```

Turn DYNCREATE off for this DBMODEL:

```
CHANGE OBJECT=DBMODEL,DIRDSN=COMPWARE.LP.SRCEDIR,DYNCREATE=NO
```

CONVERT (CNVT)

The CONVERT command first changes the internal format of an existing standard DDIO source file (specified in the DDNAME or DSN parameter) to that of a Source Listing Database format. Then it performs the ATTACH function on the file, attaching it to an existing shared directory, specified by the DIRDD or DIRDSN parameter.

The following diagram illustrates the syntax for the CONVERT command.

```
►►─CoNVerT─┬─DIRDD=ddname─┬─,─┬─DDName=ddname─┬──►◄
           └─DIRDSN=dsn───┘   └─DSN=dsn───────┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Convert the standard DDIO source file to a Source Listing Database and attach it to a shared directory.

```
CONVERT DSN=COMPWARE.SOURCE.DDIO,
        DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR
```

Note: This command will modify a record in the existing DDIO file, so it is advisable that you perform a file backup before executing this command. The source DDIO file name you specified in the command parameters will remain the same for the new database format. The CONVERT command merely changes the type and functioning of the DDIO file. If you want the source database to have a different name after it becomes attached to a shared directory, you must rename the original DDIO file prior to executing the CONVERT command.

If an error occurs during the CONVERT process in the attach phase of adding members into the shared directory, the database will set “attach in progress” (will show on a DIRX), and the shared directory MAY contain entries for only some (but not all) of the database members. If the problem cannot be solved by another attempt at the CONVERT, then perform the following to eliminate possible integrity problems:

Run UNCONVERT on the database to clear the attach setting, remove entries added to the shared directory during the process, and reset it back to a standard DDIO file format. Then run CONVERT again after determining the cause of the original errors.

COPY

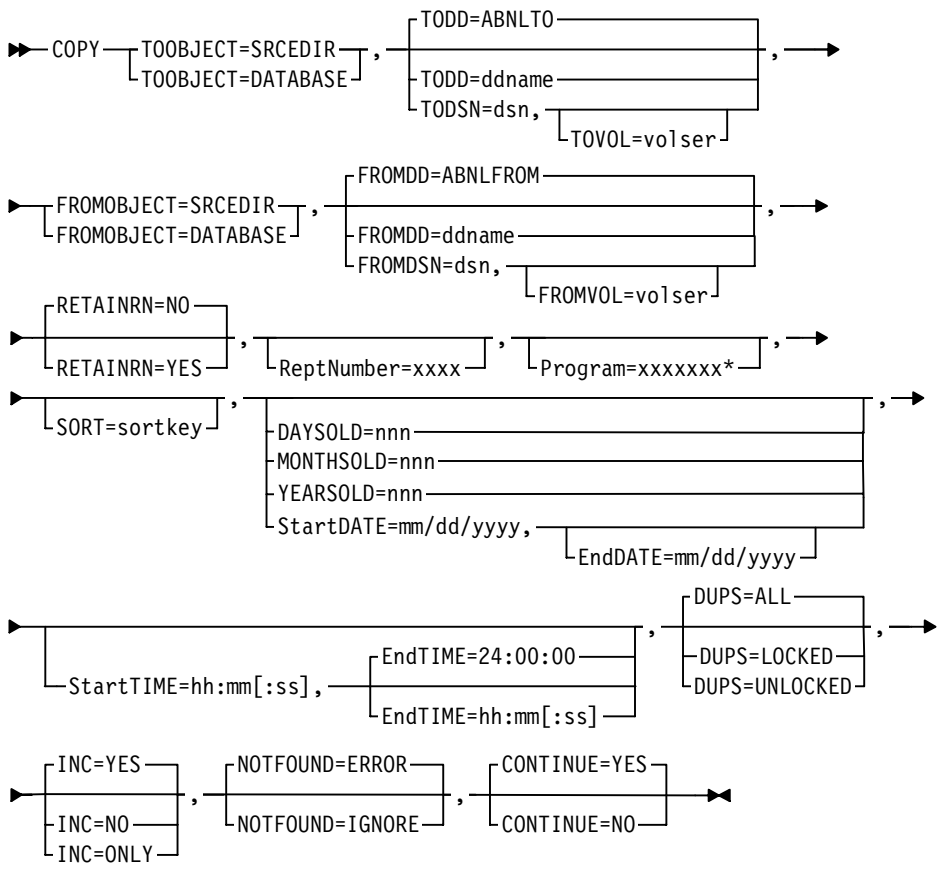
The COPY command copies selected entries in any of the following ways:

- from one Source Listing Database to another Source Listing Database
- from a Source Listing Database to a Shared Directory
- from a Shared Directory to a Source Listing Database
- from one Shared Directory to another Shared Directory

Use the FROMDD and TODD parameters to specify the “from” and “to” file ddnames. When an entry is copied to or from a Source Listing Database, which is attached to a shared directory, the shared directory is also updated immediately. If none of the parameters are specified, then all entries are copied.

Note: Since “IDR token” is a unique key within a Source Listing Shared Directory, it is possible to encounter a “duplicate IDR” condition during member COPY. A warning message will be issued, and the member will not be copied. The remaining COPY processing continues with a return code of 4.

The following diagram illustrates the syntax for the COPY command



For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Copy all entries from a Source Listing Database to another Source Listing Database.

```
COPY TOOBJECT=DATABASE,TODSN=COMPWARE.LP.SOURCE.LISTING.LPDB01,
     FROMOBJECT=DATABASE,FROMDSN=COMPWARE.LP.SOURCE.LISTING.LPDB02
```

CREATE (CRE)

The CREATE command is used to perform the following functions:

- Allocate and format a new shared directory.
- Format a pre-allocated shared directory.
- Allocate and format a new Source Listing Database.
- Format a pre-allocated Source Listing Database.
- Create a DBMODEL entry for dynamic database creation (DYNCREATE) in the specified shared directory, to be invoked during creation of a source listing member if all of the attached Source Listing Databases are full.

CAUTION:
This command erases all of the information in the specified file.

Only VSAM files can be used for shared directory files. However, either VSAM or sequential files can be used for Source Listing Databases.

When you create a Source Listing Database, it is automatically connected to the shared directory that you specify as a parameter of the CREATE command. A Source Listing Database must be connected to a shared directory before it can be used.

When you create a DBMODEL, the information is stored in the shared directory for future use in dynamic creation of a new Source Listing Database.

Sample JCL Members

Sample JCL members for the CREATE command are available in CSS installation library members. Please refer to Table 7-3 for a list of the sample members.

Table 7-3. Sample JCL Members for CREATE Command

If you want to format...	Use sample member...
Sequential Source Listing Databases	CXALLBLP
VSAM Source Listing Databases	CXALLVLP
Shared Directories	CXALLDLP

The following diagrams illustrate the syntax used for the CREATE command based on the function you want to perform.

Allocating and Formatting a New Shared Directory:

The first CREATE syntax diagram illustrates the syntax for allocating and formatting a new shared directory.

```
►►─CREate─OBJect=SRCEDIR,──┬─DIRDD=ddname──────────────┬─,─AM=VSAM,─►
                           └─DIRDSN=dsn,─┬─────────────┬─┘
                                         └─DIRVOL=volser─┘

►─┬─────────────┬─,─┬─────────────────┬─,─┬─────────────┬─,─┬─────────┬─,─►
  ├─RECords=nnnn─┤   │ ┌─CISIZE=4096─┐   │ └─VSAMCAT=ddname─┘ └─MODEL=dsn─┘
  ├─TRacKS-nnnn──┤   ├─┼─────────────┼───┤
  ├─CYLinderS=nnnn─┤   │ └─CISIZE=nnnn─┘   │
  └─DIRENTS=nnnn─┘   │ ┌─BLKsize=4089─┐   │
                     └─┼─────────────┼───┘
                       └─BLKsize=nnnn─┘

  ┌─REALLOcate=NO─┐
►─┼─────────────┼─,─┬─────────────────┬─,─┬─────────────────┬─,─►
  └─REALLOcate=YES─┘   └─MGMTCLAS=xxxxxxxx─┘   └─STORCLAS=xxxxxxxx─┘

►─┬─────────────────┬─►◄
  └─DATACLAS=xxxxxxxx─┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Create a shared directory with a CISIZE of 24569, which can occupy five cylinders.

```
CREATE OBJECT=SRCEDIR,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
       AM=VSAM,CYLINDERS=5,CISIZE=24569
```

Formatting a Pre-Allocated Shared Directory

This CREATE syntax diagram illustrates the syntax for a pre-allocated shared directory.

```
                                                  ┌─SECONDARY=NO─┐
►►─CREate─OBJect=SRCEDIR,──┬─DIRDD=ddname─┬─,──┼──────────────┼─►◄
                           └─DIRDSN=dsn───┘   └─SECONDARY=YES─┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Format a pre-allocated shared directory and specify use of secondary extents.

```
CREATE OBJECT=SRCEDIR,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
       SECONDARY=YES
```

Allocating and Formatting a New Source Listing Database

This CREATE syntax diagram illustrates the syntax for allocating and formatting a new Source Listing Database.

```
►►─CREate─OBJect=DATABASE,─┬─DIRDD=ddname─┬─(1),─┬─DDName=ddname──────────────┬─,─►
                           └─DIRDSN=dsn───┘      └─DSN=dsn,─┬─────────────────┬─┘
                                                            └─VOLser=volser───┘

►─┬─AM=VSAM,─┬──────────────┬─,─┬──────────────┬─,─┬──────────────┬─┬─,─►
  │          ├─RECords=nnnn──┤   ├─┬─CISIZE=4096─┬─┤   └─VSAMCAT=ddname─┘   │
  │          ├─TRacKS=nnnn───┤   ├─┼─────────────┼─┤                        │
  │          └─CYLinderS-nnnn┘   ├─└─CISIZE=nnnn──┘   │                        │
  │                              ├─┬─BLKsize=4089─┬─┤                        │
  │                              └─┼─────────────┼─┘                        │
  │                                └─BLKsize=nnnn─┘                          │
  │                                                                          │
  └─AM=BSAM,─┬──────────────┬─,─┬─UNIT=SYSDA───┬─,─┬─BLKsize=4096─┬─┘
             ├─TRacKS=nnnn───┤   ├─┼─────────────┼─┤  └─BLKsize=nnnn─┘
             └─CYLinders=nnnn─┘   └─UNIT=xxxxxxxx─┘   └─BLKsize=nnnn─┘

►─┬──────────┬─,─┬─REALLOcate=NO──┬─,─┬──────────────────┬─,─►
  └─MODEL=dsn─┘   └─REALLOcate=YES─┘   └─MGMTCLAS=xxxxxxxx─┘

►─┬──────────────────┬─,─┬──────────────────┬─,─┬─ReptCount=100──┬─,─►
  └─STORCLAS=xxxxxxxx─┘   └─DATACLAS=xxxxxxxx─┘   └─ReptCount=nnnn─┘

►─TYPE=SOURCE,─┬─GroupCount=4──┬─,─┬─AutoDelete=YES───┬─,─┬─EXTENTS=64──┬─►◄
               └─GroupCount=nn─┘   ├─AutoDelete=DUPS───┤   └─EXTENTS=nnn─┘
                                   ├─AutoDelete=NO─────┤
                                   └─AutoDelete=STAGED─┘
```

Note:

1. Please note that the DIRDD parameter (or DIRDSN) specifies the name of the shared directory to which you are attaching the Source Listing Database. The DDNAME parameter (or DSN) specifies the name of the Source Listing Database.

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Create a sequential Source Listing Database with the following characteristics:

- The maximum number of entries in the file is 50.
- At least one entry of each name will remain in the Source Listing Database.
- When additional space is required in the Source Listing Database, one block will be allocated.
- The maximum number of extents to be allocated to each Source Listing Database entry is 256.
- The Source Listing Database can occupy 5 cylinders.
- The blocksize is 24569.

```
CREATE OBJECT=DATABASE,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
       DSN-COMPWARE.LP.SOURCE.LISTING.LPDB01,TYPE=SOURCE,
       RC=50,AD=DUPS,GC=1,EXTENTS=256,AM=BSAM,CYLINDERS=5,
       BLKSIZE=24569
```

Formatting a Pre-Allocated Source Listing Database

This CREATE syntax diagram illustrates the syntax for formatting a pre-allocated Source Listing Database.

```
►►─CREate─OBJect=DATABASE,─┬─DIRDD=ddname─┬─(1),─┬─DDName=ddname─┬─,──►
                           └─DIRDSN=dsn───┘      └─DSN=dsn───────┘

►─┬─SECONDARY=NO──┬─,─┬─ReptCount=100─┬─,─TYPE=SOURCE,──►
  └─SECONDARY=YES─┘   └─ReptCount=nnn─┘

►─┬─GroupCount=4──┬─,─┬─AutoDelete=YES───────┬─,─┬─EXTENTS=64──┬──►◄
  └─GroupCount=nn─┘   ├─AutoDelete=DUPS──────┤   └─EXTENTS=nnn─┘
                      ├─AutoDelete=NO────────┤
                      └─AutoDelete=STAGED────┘
```

Note:

1. Please note that the DIRDD parameter (or DIRDSN) specifies the name of the shared directory to which you are attaching the Source Listing Database. The DDNAME parameter (or DSN) specifies the name of the Source Listing Database.

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Format a pre-allocated Source Listing Database with the following characteristics:

- The maximum number of entries in the file is 100.
- Autodelete entries when full.
- When additional space is required in the Source Listing Database, two blocks will be allocated.
- The maximum number of extents to be allocated to each Source Listing Database entry is 64.

```
CREATE OBJECT=DATABASE,
       DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
       DSN=COMPWARE.LP.SOURCE.LISTING.LPDB01,TYPE=SOURCE,
       RC=100,AD=YES,GC=2,EXTENTS=64
```

Creating a DBMODEL

This CREATE syntax diagram illustrates the syntax for creating a DBMODEL entry.

```
►►─CREate──OBJect=DBMODEL,──┬─DIRDD=ddname──┬─(1),──DSN=dbmodelname(2),──►
                            └─DIRDSN=dsn────┘

►─┬─AM=VSAM,─┬──────────────┬─,─┬──────────────────┬─,─┬──────────────────┬─┬─,─►
  │          ├─RECords=nnnn──┤   │ ┌─CISIZE=4096─┐  │   └─VSAMCAT=ddname───┘ │
  │          ├─TRacKS=nnnn───┤   ├─┼─────────────┼──┤                        │
  │          └─CYLinderS=nnnn┘   │ └─CISIZE=nnnn─┘  │                        │
  │                              │ ┌─BLKsize=4089─┐ │                        │
  │                              ├─┼──────────────┼─┤                        │
  │                              │ └─BLKsize=nnnn─┘ │                        │
  │                              ┌─UNIT=SYSDA────┐  ┌─BLKsize=4096──┐        │
  └─AM=BSAM,─┬──────────────┬─,─┼──────────────┼─,─┼───────────────┼────────┘
             ├─TRacKS=nnnn───┤   └─UNIT=xxxxxxxx─┘  └─BLKsize=nnnn───┘
             └─CYLinders=nnnn┘

►─┬──────────┬─,─┬──────────────────┬─,─┬──────────────────┬─,─┬──────────────────┬─,─►
  └─MODEL=dsn─┘   └─MGMTCLAS=xxxxxxxx─┘   └─STORCLAS=xxxxxxxx─┘   └─DATACLAS=xxxxxxxx─┘

  ┌─ReptCount=100──┐                ┌─GroupCount=4──┐  ┌─AutoDelete=YES───┐
►─┼────────────────┼─,─TYPE=SOURCE,─┼───────────────┼─,─┼──────────────────┼─,─►
  └─ReptCount=nnnn─┘                └─GroupCount=nn─┘  ├─AutoDelete=DUPS───┤
                                                       ├─AutoDelete=NO─────┤
                                                       └─AutoDelete=STAGED─┘

  ┌─EXTENTS=64──┐  ┌─DYNCREATE=NO──┐
►─┼─────────────┼─,─┼───────────────┼─►◄
  └─EXTENTS=nnn─┘  └─DYNCREATE=YES─┘
```

Notes:

1. Please note that the DIRDD parameter (or DIRDSN) specifies the name of the shared directory to which you are attaching the Source Listing Database. The DDNAME parameter (or DSN) specifies the name of the Source Listing Database.
2. Please note that the “dbmodelname” used here specifies the name to be created for the new Source Listing Database. It consists of literals and/or symbolic tokens that will generate the final DSN name at creation time. Remember, the total of generated characters plus separator periods must not exceed 44. See the DSN parameter description for more detail.

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Create a DBMODEL for a sequential Source Listing Database with the following characteristics:

- The maximum number of entries in the file is 50.
- At least one entry of each name will remain in the Source Listing Database.
- When additional space is required in the Source Listing Database, one block will be allocated.
- The maximum number of extents to be allocated to each Source Listing Database entry is 256.
- The Source Listing Database can occupy 5 cylinders.
- The blocksize is 24569.
- Dynamic database creation will be invoked if no attached databases are eligible for create.
- The new DSN name will have an incremental sequence number assigned.

```
CREATE OBJECT=DBMODEL,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
       DSN=COMPWARE.LP.SOURCE.LISTING.&SEQ,DYNCREATE=YES,
       RC=50,AD=DUPS,GC=1,EXTENTS=256,AM=BSAM,CYLINDERS=5,
       BLKSIZE=24569
```

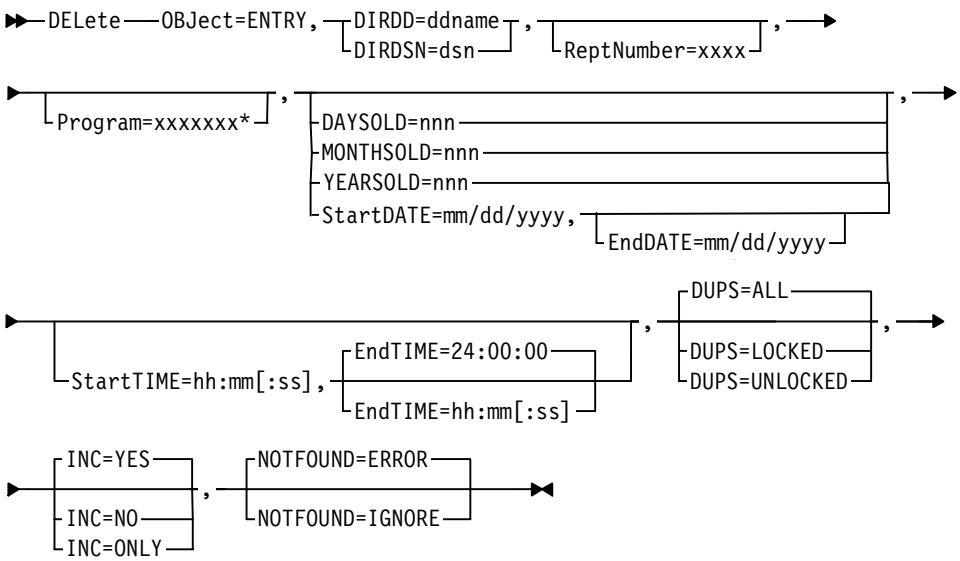
DELETE (DEL)

The DELETE command deletes selected Source Listing Database entries, a selected Source Listing Database, a selected shared directory, or the DBMODEL entry information contained in the selected shared directory.

Note: The DELETE command, while it removes records, may or may not free up any of the allocated blocks in use depending on whether other data are sharing the same blocks as the records being deleted.

DELETE an ENTRY from Both the Shared Directory and Source Listing Database

DELETE removes the entry from both the shared directory and the Source Listing Database. Source Listing Database entries can only be deleted if attached to a shared directory. You must specify either the DIRDD or the DIRDSN parameter. (To delete an entry, specify OBJECT=ENTRY.)



For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Delete all entries with a name beginning with ‘COBA’:

```
DELETE OBJECT=ENTRY,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
       PROGRAM=COBA*
```

DELETE a Source Listing Database (and Detach It from the Shared Directory)

If the Source Listing Database is attached to a shared directory, DELETE also detaches it from the shared directory prior to deletion. You must specify either the DDNAME or the DSN parameter. (To delete a Source Listing Database, specify OBJECT=DATABASE.)

```
►►─DELete──OBJect=DATABASE,──┬─DDName=ddname──────────────────┬──►◄
                             └─DSN=dsn,──┬──────────────────┬─┘
                                         └─VOLser=volser────┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Delete a Source Listing Database (and detach it from the shared directory).

```
DELETE OBJECT=DATABASE,
       DSN=COMPWARE.LP.SOURCE.LISTING.DB01
```

DELETE a Shared Directory

When deleting a shared directory, you can choose whether to also delete all attached Source Listing Databases. Regardless of your choice, a DETACH is performed on all attached Source Listing Databases prior to the deletion. You must specify either the DIRDD or the DIRDSN parameter. (To delete a shared directory, specify OBJECT=SRCEDIR.)

```
►►─DELete─OBJect=SRCEDIR,──┬─DIRDD=ddname─┬──,──►
                           └─DIRDSN=dsn───┘
   ┌─DataBaseDISP=DETACH─┐
►──┼─────────────────────┼──────────────────►◄
   └─DataBaseDISP=DELETE─┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Examples

Delete a shared directory and any Source Listing Databases attached to it.

```
DELETE OBJECT=SRCEDIR,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
       DBDISP=DELETE
```

Delete a shared directory, but first detach any Source Listing Databases attached to it (does NOT delete databases).

```
DELETE OBJECT=SRCEDIR,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
       DBDISP=DETACH
```

DELETE the DBMODEL Information from the Shared Directory

```
►►─DELete─OBJect=DBMODEL,──┬─DIRDD=ddname─┬──►◄
                           └─DIRDSN=dsn───┘
```

For a description of each parameter, see "Command Parameters for Shared Directories and Source Listing Databases" on page 7-6.

Example

Delete a DBMODEL:

```
DELETE OBJECT=DBMODEL,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR
```

DETACH

The DETACH command disconnects a Source Listing Database from a shared directory and removes the Source Listing Database entries from the shared directory.

To ensure that you are aware of the shared directory from which you are detaching the Source Listing Database, you can optionally use the DIRDSN parameter to specify the name of the shared directory. If the shared directory is incorrectly specified, an error will be returned. If the DIRDSN is not specified, then the Source Listing Database will be unconditionally detached from the appropriate shared directory.

The following diagram illustrates the syntax for the DETACH command.

```
►►─DETACH──┬─DDName=ddname──────────────┬─┬──────────────┬──►◄
           └─DSN=dsn,──┬──────────────┬─┘ └─DIRDSN=dsn─┘
                       └─VOLser=volser─┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Detach a Source Listing Database from a shared directory.

```
DETACH DDNAME=LPFILE1
```

Note: If an error occurs during the DETACH process, when deleting members from the shared directory, the database will set “detach in progress” (will show on a DIRX) but will still appear attached, and the shared directory may still contain entries for members on the database that had not yet been removed from the shared directory. To eliminate possible integrity problems perform the following:

1. Run RESET on the database to clear the attach setting.
2. Run SCRATCH on the shared directory for that database to remove any entries left for that database.

DIRECTORY (DIR, D) and DIRX (DX)

The DIRECTORY command lists selected entries in a shared directory or Source Listing Database.

The function of the DIRX command is similar to the DIRECTORY command, but it also displays attribute and allocation information. When a DIRX is performed on a shared directory, it displays the attribute and allocation information for the shared directory and all Source Listing Databases attached to that shared directory.

The following diagram illustrates the syntax for the DIRECTORY or DIRX command.

```
►►─DIRectory or DIRX─┬──────────────────────┬─,──►
                     └ReptNumber=xxxxxxxxxx─┘

►──┬─OBJect=SRCEDIR,──┬─DIRDD=ddname─┬──────────────────┬─,──►
   │                  └─DIRDSN=dsn───┘                  │
   └─OBJect=DATABASE,──┬─DDName=ddname──────────────────┬─┘
                       └─DSN=dsn,──┬──────────────────┬─┘
                                   └─VOLser=volser────┘

►──┬──────────────────┬─,──────────►
   └─Program=xxxxxxx*─┘

►──┬──────────────────────────────────────────────────┬─,──►
   ├─DAYSOLD=nnn──────────────────────────────────────┤
   ├─MONTHSOLD=nnn────────────────────────────────────┤
   ├─YEARSOLD=nnn─────────────────────────────────────┤
   └─StartDATE=mm/dd/yyyy,──┬──────────────────────┬──┘
                            └─EndDATE=mm/dd/yyyy───┘

►──┬──────────────────────────────────────────────────┬─,──►
   │                      ┌─EndTIME=24:00:00──────┐   │
   └─StartTIME=hh:mm[:ss],─┼──────────────────────┼───┘
                          └─EndTIME=hh:mm[:ss]────┘

   ┌─DUPS=ALL──────────┐   ┌─INC=YES──────┐   ┌─NOTFOUND=ERROR──────┐
►──┼───────────────────┼─,─┼──────────────┼─,─┼─────────────────────┼─,──►
   ├─DUPS=LOCKED───────┤   ├─INC=NO───────┤   └─NOTFOUND=IGNORE────┘
   └─DUPS=UNLOCKED─────┘   └─INC=ONLY─────┘

►──┬──────────────────┬──►◄
   └─SORT=sortkey─────┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6. Refer to Shared Services SLXCNTL installation library member CXLPDIRX for sample JCL for the DIRX command.

Example 1

List all entries in a shared directory.

```
DIR OBJECT=SHRDIR,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR
```

Example 2

List all entries in a Source Listing Database sorted by program name (PGM), and include attribute and allocation information.

```
DIRX OBJECT=DATABASE,DSN=COMPWARE.LP.SOURCE.LISTING.LPDB01,
SORT=PGM
```

DIRECTORY SEQUENTIAL (DIRS)

The DIRS (DIRectory Sequential) command lists selected directory entries in a sequential file produced by the EXPORT command. If none of the parameters are specified, then all entries are listed.

The following diagram illustrates the syntax for the DIRECTORY SEQUENTIAL command.

```
►►─DIRS──┬─DDName=ddname──────────────┬─,──►
         └─DSN=dsn,──┬──────────────┬─┘
                     └─VOLser=volser─┘
►─┬──────────────────┬─,─┬──────────────┬─,──►
  └─ReptNumber=xxxx──┘   └─Program=xxxxxxx*─┘

►─┬──────────────────────────────────────────────┬─,──►
  ├─DAYSOLD=nnn──────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,──┬──────────────────┬─┘
                           └─EndDATE=mm/dd/yyyy─┘

►─┬──────────────────────────────────────────────┬─,──►
  │                          ┌─EndTIME=24:00:00──┐  │
  └─StartTIME=hh:mm[:ss],──┼──────────────────┼─┘
                           └─EndTIME=hh:mm[:ss]──┘

  ┌─DUPS=ALL──────┐   ┌─INC=YES──┐   ┌─NOTFOUND=ERROR──┐
►─┼───────────────┼─,─┼──────────┼─,─┼─────────────────┼─,──►
  ├─DUPS=LOCKED───┤   ├─INC=NO───┤   └─NOTFOUND=IGNORE──┘
  └─DUPS=UNLOCKED─┘   └─INC=ONLY─┘

                        ┌─LABEL=1──┐
►─┬──────────────┬─,─┼──────────┼──►◄
  └─SORT=sortkey─┘   └─LABEL=nn──┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Export all unlocked entries and list all directory entries on the resulting sequential file.

```
EXPORT DUPS=UNLOCKED
DIRS DSN=EFHBRR0.EXFILE1
```

ESTIMATE

The ESTIMATE command is used to determine a reasonable number of blocks to allocate for a shared directory when the allocation is being performed outside of CSS (i.e., through IDCAMS or Compuware's File-AID product). The resulting calculation is expressed in records.

The following diagram illustrates the syntax used for the ESTIMATE command.

```
►►─ESTIMATE─OBJect=SRCEDIR,──┬─BLKsize=nnnn─┐ ,──┬──────────────┬──►◄
                             ├─CISIZE=nnnn──┤    └─DIRENTS=nnnn──┘
                             └─MODEL=dsn────┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Estimate the number of blocks for a shared directory with CISIZE=24569 and approximately 200 directory entries.

```
ESTIMATE OBJECT=SRCEDIR,CISIZE=24569,DIRENTS=200
```

EXPAND

The EXPAND command reallocates an existing shared directory using new allocation parameters and copies all of the entries to the newly allocated shared directory. After the copy has completed successfully, the old shared directory is renamed to a temporary name, the new shared directory is renamed to the original name, and the old shared directory is then deleted. You can use this command to increase or decrease the space allocated for a shared directory and/or change the CISIZE.

When using the EXPAND command, the following rules apply:

- The shared directory must already exist (i.e., you cannot use EXPAND to create a new shared directory).
- The shared directory must not be in use.

Use the DIRDD or DIRDSN parameter to specify the existing shared directory. You can optionally use the VOLSER to specify the desired volume location for the new expanded shared directory.

The following diagram illustrates the syntax for the EXPAND command.

```
>>-EXPAND--OBJect=SRCEDIR,--+DIRDD=ddname--+,--+-------------+,-->
                            +DIRDSN=dsn---+    +VOLser=volume+
                                                 +CISIZE=4096-+
>--+-------------+,--+-------------+,--+-------------+,-->
   +VSAMCAT=ddname+   +RECords=nnnn--+   +CISIZE=nnnn-+
                      +TRacKS=nnnn---+
                      +CYLinders=nnnn+
>--+-------------+-->
   +MODEL=dsn-+
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Expand an existing shared directory, but change the CISIZE to 8192.

```
EXPAND OBJECT=SRCEDIR,DIRDD=CAT11,
       MODEL=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,CISIZE=8192
```

EXPORT

The EXPORT command exports selected entries from a Shared Directory or Source Listing Database to a sequential file. The sequential file can then be used as input to the IMPORT command.

The “from” file (the Shared Directory or Source Listing Database) can be specified using either the FROMDSN or FROMDD parameter.

The “to” file (the sequential file) can be specified using either the TODSN and TOVOL parameters, or by using the TODD parameter. However, if you are exporting to a tape, you must use the TODD parameter and not the TODSN and TOVOL parameters. (The ABNLFROM and ABNLTO DD statements can be used for exporting.) When an entry is exported from a Source Listing Database and DELETE=YES, the shared directory is also updated.

If none of the parameters are specified, then all entries are exported. You may use multiple EXPORT commands if you include a unique TODD (or TODSN and TOVOL) parameter for each output dataset or PDS member that is a target of the export. If unique parameters are not used, each successive export will overlay the previous export.

The following diagram illustrates the syntax for the EXPORT command.

```
►►─EXPORT──┬─FROMOBJECT=SRCEDIR──┬─,──►
           └─FROMOBJECT=DATABASE─┘

►─┬─FROMDD=ABNLFROM──────────────┬─,─┬─TODD=ABNLTO──────────────┬─,──►
  ├─FROMDD=ddname ───────────────┤   ├─TODD=ddname──────────────┤
  └─FROMDSN=dsn,──┬─────────────┘   └─TODSN=dsn,──┬─────────────┘
                  └─FROMVOL=volser─┘                └─TOVOL=volser─┘

►──┬─────────────────┬─,──┬─────────────────┬─,──►
   └─ReptNumber=xxxx─┘    └─Program=xxxxxxx*─┘

►─┬─────────────────────────────────────────────────┬─,──►
  ├─DAYSOLD=nnn──────────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,──┬─────────────────────────┘
                           └─EndDATE=mm/dd/yyyy─┘

►─┬─────────────────────────────────────────────────┬─,──►
  │                          ┌─EndTIME=24:00:00──┐   │
  └─StartTIME=hh:mm[:ss],──┬─────────────────────┬─┘
                           └─EndTIME=hh:mm[:ss]──┘

►─┬─DUPS=ALL──────────┬─,──┬─INC=YES──────┬─,──┬─NOTFOUND=ERROR──────┬─,──►
  ├─DUPS=LOCKED───────┤    ├─INC=NO───────┤    └─NOTFOUND=IGNORE───────┘
  └─DUPS=UNLOCKED─────┘    └─INC=ONLY─────┘

►─┬─CONTINUE=YES──┬─,──┬─LABEL=1──────┬─,──┬─DELETE=NO──────┬──►◄
  └─CONTINUE=NO───┘    └─LABEL=nn─────┘    └─DELETE=YES─────┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6. Refer to Shared Services SLXCNTL installation library member CXLPEXPO for sample JCL for the EXPORT command.

Example

For a Source Listing Database, export all complete source listings for the program names starting with ABCD and compiled on May 1, 2000 to a tape. Delete all entries that have been successfully exported.

```
EXPORT FROMOBJECT=DATABASE,FROMDD=ABNLFROM,
       TODD=ABNLTO,
       PROGRAM=ABCD*,STARTDATE=05/01/2000,
       INC=NO,DELETE=YES
```

IMPORT

The IMPORT command selectively restores previously exported entries from a sequential file to a shared directory or source listing database.

The “from” file (the sequential file) can be specified using either the FROMDSN and FROMVOL parameters, or by using the FROMDD parameter. The “from” file can also be a file created by the batch DDIO utility CWDDSUTL EXPORT command for a DDIO source file.

The “to” file (the Shared Directory or Source Listing Database) can be specified using either the TODSN and TOVOL parameters, or by using the TODD parameter. When an entry is imported to a Source Listing Database that is attached to a shared directory, the shared directory is also updated.

Note: If the access method is VSAM, then you must specify DISP=SHR for the TODSN DD statement in your JCL.

Since “IDR token” is a unique key within a Source Listing Shared Directory, it is possible to encounter a “duplicate IDR” condition during member IMPORT – a warning message will be issued and the member will not be imported. The remaining IMPORT processing continues with a return code of 4.

If none of the parameters are specified, then all entries are imported. The following diagram illustrates the syntax for the IMPORT command.

```
►►─IMPORT──┬─TOOBJECT=SRCEDIR──┬─,──►
           └─TOOBJECT=DATABASE─┘

►─┬─FROMDD=ABNLFROM──────────┬─,─┬─TODD=ABNLTO──────────┬─,──►
  ├─FROMDD=ddname ───────────┤   ├─TODD=ddname──────────┤
  └─FROMDSN=dsn,─┬─────────┬─┘   └─TODSN=dsn,─┬─────────┬─┘
                 └─FROMVOL=volser┘                └─TOVOL=volser┘

►─┬─────────────┬─,─┬─────────────┬─,──►
  └─ReptNumber=xxxx┘  └─Program=xxxxxxx*─┘

►─┬─────────────────────────────────────────┬─,──►
  ├─DAYSOLD=nnn──────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,─┬─────────────┬─┘
                          └─EndDATE=mm/dd/yyyy─┘

►─┬─────────────────────────────────────────┬─,─┬─DUPS=ALL──────┬─,──►
  │                                         │   ├─DUPS=LOCKED────┤
  └─StartTIME=hh:mm[:ss],─┬─EndTIME=24:00:00─┬─┘   └─DUPS=UNLOCKED─┘
                          └─EndTIME=hh:mm[:ss]─┘

►─┬─INC=YES──┬─,─┬─NOTFOUND=ERROR──┬─,─┬─CONTINUE=YES─┬─,─┬─LABEL=1──┬──►◄
  ├─INC=NO───┤   ├─NOTFOUND=IGNORE─┘   └─CONTINUE=NO──┘   └─LABEL=nn──┘
  └─INC=ONLY─┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6. Refer to Shared Services SLXCNTL installation library member CXLPIMPO for sample JCL for the IMPORT command.

Example

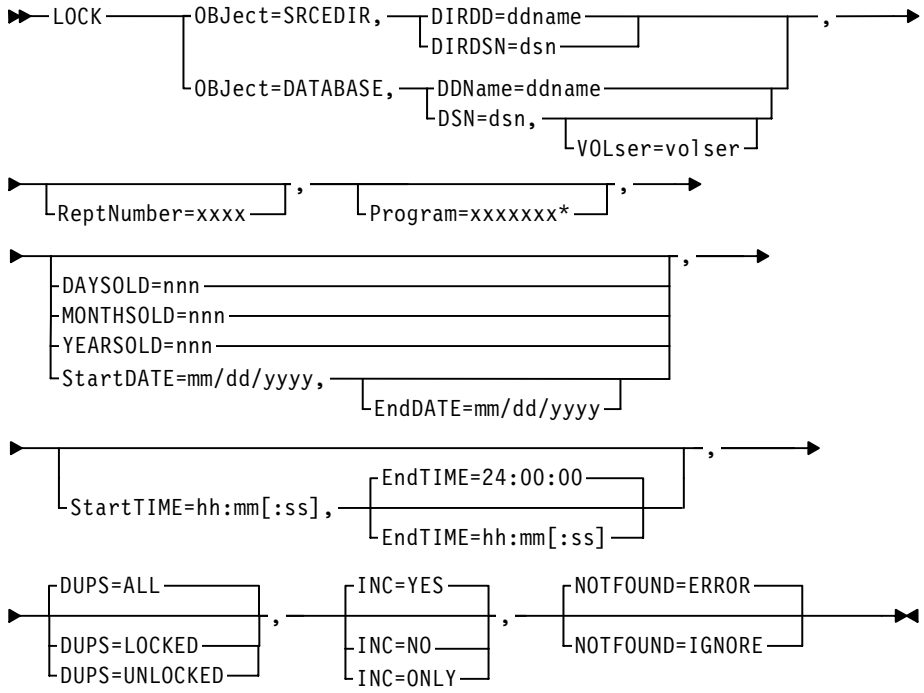
Restore all entries from a sequential file to a Source Listing Database.

```
IMPORT TOOBJECT=DATABASE,FROMDSN=EFHBMM0.DB01,
       TODSN=COMPWARE.LP.SOURCE.LISTING.LPDB01
```

LOCK

The LOCK command locks selected entries of a specified Source Listing Database or all Source Listing Databases attached to a specified shared directory. While an entry is locked with this command (i.e., manually locked), it cannot be automatically deleted when the Source Listing Database becomes full.

The following diagram illustrates the syntax for the LOCK command.



For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

For a Source Listing Database, lock source listing number 1105.

```
LOCK OBJECT=DATABASE,
     DSN=COMPWARE.LP.SOURCE.LISTING.LPDB01,REPTNUMBER=1105
```

MOVE

The MOVE command copies selected entries from one Shared Directory or Source Listing Database to another Shared Directory or Source Listing Database and deletes them from the “from” file, provided they are successfully copied. Entries can be moved in any of the following ways:

- from one Source Listing Database to another Source Listing Database
- from a Source Listing Database to a Shared Directory
- from a Shared Directory to a Source Listing Database
- from one Shared Directory to another Shared Directory

Use the FROMDD and TODD parameters to specify the “from” and “to” file ddnames. When an entry is moved to or from a Source Listing Database that is attached to a shared directory, the shared directory is also updated. If none of the parameters are specified, then all entries are moved.

Note: Only successfully copied entries are deleted from the “from” file.

Since “IDR token” is a unique key within a Source Listing Shared Directory, it is possible to encounter a “duplicate IDR” condition during member MOVE – a warning message will be issued and the member will not be moved. The remaining MOVE processing continues with a return code of 4.

The following diagram illustrates the syntax for the MOVE command.

```
                                   ┌TODD=ABNLTO──────────────┐
►►─MOVE─┬TOOBJECT=SRCEDIR─┬,─┼──────────────────────────┼,─►
        └TOOBJECT=DATABASE┘   ├TODD=ddname──────────────┤
                              └TODSN=dsn,─┬──────────────┘
                                          └TOVOL=volser┘
                                ┌FROMDD=ABNLFROM──────────────┐   ┌RETAINRN=NO─┐
►──┬FROMOBJECT=SRCEDIR─┬,─┼──────────────────────────┼,─┼────────────┼,─►
   └FROMOBJECT=DATABASE┘   ├FROMDD=ddname──────────────┤   └RETAINRN=YES┘
                           └FROMDSN=dsn,─┬──────────────┘
                                         └FROMVOL=volser┘

►──┬──────────────┬,─┬──────────────┬,─┬──────────────┬,─►
   └ReptNumber=xxxx┘   └Program=xxxxxxx*┘   └SORT=sortkey┘

►──┬──────────────────────────────────────────────────┬─,─►
   ├DAYSOLD=nnn──────────────────────────────────────┤
   ├MONTHSOLD=nnn────────────────────────────────────┤
   ├YEARSOLD=nnn─────────────────────────────────────┤
   └StartDATE=mm/dd/yyyy,─┬──────────────────────────┘
                          └EndDATE=mm/dd/yyyy┘
                                                   ┌DUPS=ALL─────┐
►──┬──────────────────────────────────────────────┬,─┼─────────────┼,─►
   │                                              │   ├DUPS=LOCKED──┤
   └StartTIME=hh:mm[:ss],─┬EndTIME=24:00:00──┬─┘   └DUPS=UNLOCKED┘
                          └EndTIME=hh:mm[:ss]┘

   ┌INC=YES──┐   ┌NOTFOUND=ERROR──┐   ┌CONTINUE=YES─┐
►──┼─────────┼,─┼───────────────┼,─┼───────────────┼──►◄
   ├INC=NO───┤   └NOTFOUND=IGNORE┘   └CONTINUE=NO──┘
   └INC=ONLY─┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6. Refer to Shared Services SLXCNTL installation library member CXLPMOVE for sample JCL for the MOVE command.

Examples

Move all entries from one shared directory to another, and do not retain the report numbers.

```
MOVE TOOBJECT=SPCEDIR,TODSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
     FROMOBJECT=SRCEDIR,FROMDSN=COMPWARE.LP.SOURCE.LISTING2.SRCEDIR,
     RETAINRN=NO
```

Move all unlocked entries from one Source Listing Database to another Source Listing Database.

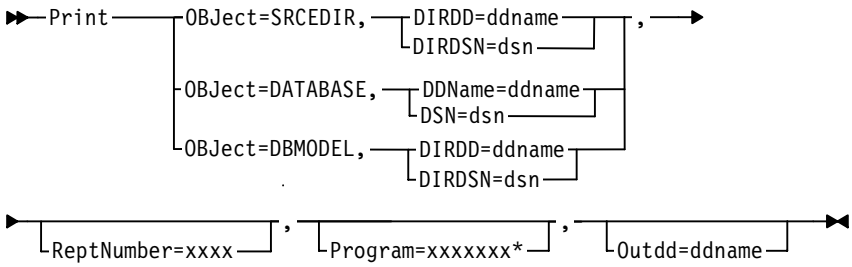
```
MOVE TOOBJECT=DATABASE,TODSN=COMPWARE.LP.SOURCE.LISTING.LPDB02,
     FROMOBJECT=DATABASE,FROMDSN=COMPWARE.LP.SOURCE.LISTING.LPDB01
```

PRINT (P)

The PRINT command prints a selected member from the specified Source Listing Database or shared directory. It also prints the specifications for a previously-defined DBMODEL. If the PROGRAM parameter is used, only the most recent source member will be printed if there are multiple entries for that name.

If the OUTDD parameter is omitted, the output (along with other CWDDLPUT messages and reports) is sent to the file specified by the ABNLREPT DD statement. If the OUTDD parameter is coded, the output is sent to the file with the ddname specified by the OUTDD parameter.

The following diagram illustrates the syntax for the PRINT command.



For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Examples

Print member 96 from a Source Listing Database.

```
PRINT OBJECT=DATABASE,DSN=COMPWARE.LP.SOURCE.LISTING.LPDB01,RN=96
```

Print DBMODEL specifications.

```
PRINT OBJECT=DBMODEL,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
```

Print member 10 from a Source Listing Shared Directory, and send the member print output to the file specified by the SRCREPT DD statement.

```
PRINT OBJECT=SRCEDIR,RN=10,OUTDD=SRCREPT
```

RESET

The RESET command disconnects a Source Listing Database from a shared directory when the shared directory is lost or damaged. The Source Listing Database can then be attached to another shared directory.

The following diagram illustrates the syntax for the RESET command.

```
►►--RESET──┬─DDName=ddname─┬──────────►◄
           └─DSN=dsn───────┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Disconnect a Source Listing Database from a damaged shared directory by resetting all the flags.

```
RESET DSN=COMPWARE.LP.SOURCE.LISTING.LPDB01
```

SCRATCH (SCR)

The SCRATCH command clears all entries from a shared directory for a specified Source Listing Database that has been lost or damaged.

The following diagram illustrates the syntax for the SCRATCH command.

```
►►─SCRatch──┬─DIRDD=ddname──┬──,──DSN=dsn──►◄
            └─DIRDSN=dsn────┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Scratch all entries from a shared directory for a lost Source Listing Database.

```
SCRATCH DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
        DSN=COMPWARE.LP.SOURCE.LISTING.LPDB01
```

TEST

The TEST command allows testing of the DBMODEL specification and “dbmodelname” previously defined for the specified shared directory. The TEST request will create and format a Source Listing Database according to the DBMODEL specifications but will NOT attach it to the shared directory.

The following diagram illustrates the syntax for the TEST command.

```
▶▶─TEST──OBJect=DBMODEL,──┬─DIRDD=ddname──┬──▶◀
                          └─DIRDSN=dsn────┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Test the allocation and formatting of the DBMODEL previously created for a shared directory.

```
TEST OBJECT=DBMODEL,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR
```

UNCONVERT (UNCNVT)

The UNCONVERT command first detaches a Source Listing Database from its shared directory, and then changes the format of the Source Listing Database back to that of a standard DDIO source listing file. Long name members will still be in the DDIO source listing file but will not correctly identify themselves and will not be usable.

To ensure that you are aware of the shared directory from which you are detaching the Source Listing Database, you can optionally use the DIRDSN parameter to specify the name of the shared directory. If the shared directory is incorrectly specified, an error will be returned. If the DIRDSN is not specified, then the Source Listing Database will be unconditionally detached from the appropriate shared directory before changing the format.

The following diagram illustrates the syntax for the UNCONVERT command.

```
►►─UNCoNVerT──┬─DIRDD=ddname─┬─,─┬─DDName=ddname─┬──►◄
              └─DIRDSN=dsn───┘   └─DSN=dsn────────┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Change the Source Listing Database back to a standard DDIO source listing file. It will automatically be detached from its shared directory.

```
UNCONVERT DSN=COMPWARE.SOURCE.DDIO.DB
```

The source listing database name you specified in the command parameter will remain the same when it reverts back to the standard DDIO source listing file format. The UNCONVERT command merely changes the type and functioning of the source listing database back to the old style format. If you want the source DDIO file to have a different name after it becomes detached from the shared directory, you must rename it after executing the UNCONVERT command.

If an error occurs during the UNCONVERT process, while in the detach phase of deleting members from the shared directory, the database will set “detach in progress” (will show on a DIRX) but will still appear attached, and the shared directory may still contain entries for members in the database that had not yet been removed from the shared directory. To eliminate possible integrity problems, perform the following:

1. Run RESET on the database to clear the attach setting.
2. Run SCRATCH on the shared directory for that database to remove any entries left for that database.
3. Run ATTACH on the database to re-add to the shared directory.
4. Run UNCONVERT again.

UNLOCK (UNLK)

The UNLOCK command unlocks selected entries of a specified Source Listing Database or all Source Listing Databases attached to a specified shared directory. When an entry is unlocked, it can be automatically deleted when the Source Listing Database becomes full if the Source Listing Database has been formatted with AUTODELETE=YES or AUTODELETE=DUPS. At least one parameter must be specified.

The following diagram illustrates the syntax for the UNLOCK command.

```
►►─UNLocK──┬─OBJect=SRCEDIR,──┬─DIRDD=ddname──┬──────────────┬─,──►
           │                  └─DIRDSN=dsn────┘              │
           └─OBJect=DATABASE,──┬─DDName=ddname──────────────┬─┘
                               └─DSN=dsn,──┬─────────────┬─┘
                                           └─VOLser=volser─┘

►─┬──────────────────┬─,──┬──────────────────┬─,──►
  └─ReptNumber=xxxx──┘    └─Program=xxxxxxx*─┘

►─┬──────────────────────────────────────────────────┬─,──►
  ├─DAYSOLD=nnn──────────────────────────────────────┤
  ├─MONTHSOLD=nnn────────────────────────────────────┤
  ├─YEARSOLD=nnn─────────────────────────────────────┤
  └─StartDATE=mm/dd/yyyy,──┬──────────────────┬──────┘
                           └─EndDATE=mm/dd/yyyy─┘

►─┬──────────────────────────────────────────────────┬─,──►
  │                          ┌─EndTIME=24:00:00──┐   │
  └─StartTIME=hh:mm[:ss],──┼──────────────────┼──┘
                             └─EndTIME=hh:mm[:ss]──┘

  ┌─DUPS=ALL──────┐      ┌─INC=YES──┐      ┌─NOTFOUND=ERROR──┐
►─┼───────────────┼─,──┼──────────┼─,──┼─────────────────┼──►◄
  ├─DUPS=LOCKED───┤      ├─INC=NO───┤      └─NOTFOUND=IGNORE──┘
  └─DUPS=UNLOCKED─┘      └─INC=ONLY─┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

Unlock entries in Source Listing Databases attached to a shared directory that are older than six months.

```
UNLOCK OBJECT=SRCEDIR,DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR,
       MONTHSOLD=6
```

VERIFY (VER)

The VERIFY command examines a selected shared directory to ensure the following:

- All Source Listing Databases attached to the Shared Directory exist.
- All entries in the Shared Directory exist in the attached Source Listing Databases and names and IDR tokens match.
- All entries in the attached Source Listing Databases are also in the Shared Directory and names and IDR tokens match.
- Shared Directory internal integrity of member numbers.
- Shared Directory internal integrity of IDR tokens.

Note: This command creates a report of any inconsistencies but does not perform any maintenance or repairs.

The following diagram illustrates the syntax for the VERIFY command.

```
►►─VERify──┬─DIRDD=ddname─┬──────────────►◄
           └─DIRDSN=dsn───┘
```

For a description of each parameter, see “Command Parameters for Shared Directories and Source Listing Databases” on page 7-6.

Example

For a shared directory, verify that all attached Source Listing Databases exist and that the entries in the Source Listing Database match the entries in the shared directory.

```
VERIFY DIRDSN=COMPWARE.LP.SOURCE.LISTING.SRCEDIR
```

Chapter 8.
Compuware Viewing Facility

The Compuware Viewing Facility (Compuware/VF) lets you view, print, lock, unlock, and delete members in DDIO files online. If you are an Abend-AID user and have Distributed Viewing Support available at your site, which requires that Base Services and Host Communications Interface (HCI) be installed, you can view reports located on a different or remote MVS system. While viewing an Abend-AID report, you can directly access any available section of the report, access Compuware File-AID, or associate contact information with a particular job.

Note: To access File-AID from Compuware/VF, one of the following File-AID releases must be installed: File-AID/MVS 8.5 or later, File-AID for DB2 V3R9 or later, or File-AID for IMS 4.7 with product update F1. See the *ECC Installation and Customization Guide* for complete installation requirements for Compuware/VF.

The following points are important to note:

- When viewing an Abend-AID report, you should use a CSS release that is the same as or more current than the CSS release that was used to write the report or source listing. Using an older CSS release at view-time may cause unpredictable results.
- Compuware/VF cannot be used to view or maintain shared directories or their associated transaction databases relating to Abend-AID for CICS.
- To get help at any point while using Compuware/VF, press PF1.

Abend-AID Primary Menu

After invoking Compuware/VF, the Abend-AID Primary Menu appears (Figure 8-1).

Figure 8-1. Abend-AID Primary Menu

```
----------------------- Abend-AID Primary Menu ----------------  CSS V8.0.0
 OPTION  ===>



   1  VIEW       - View Abend-AID reports or Source listings
   2  CREATEJCL  - Create JCL to invoke Compuware Language Processor (LP)
   3  SOURCE     - Create/Format Source listing files(s)/SHRDIR/Database(s)
   4  AAREPORT   - Create/Format Abend-AID Report file(s)/SHRDIR/DB(s)
   5  CONTACTS   - Update Contact Information


   L  LP         - Learn about Compuware Language Processor
   D  DDIO       - Learn about Source Listing and Report DDIO files
   C  LEGAL      - Display Full Copyright/Trade Secrets Notice

     Copyright (c) 1976-2004, Compuware Corporation. All rights Reserved.
     Unpublished - rights reserved under the Copyright Laws of the United
     States.
                            (800) 538-7822
                        Compuware Shared Services

            Press ENTER to process or enter END command to terminate
```

To choose an option, enter the corresponding letter or number at the OPTION prompt, or just press **Enter** to select the default option, View. The primary menu options are:

Table 8-1. Abend-AID Primary Menu Options

Menu Option	Does This	For More Information
VIEW	Displays the Compuware/VF Entry Panel	See "Compuware/VF Entry Panel" (Figure 8-2)
CREATEJCL	Creates JCL to run the Compuware preprocessors or postprocessors.	See "CREATEJCL - Preparing Compiler Listings for the LP" on page 3-4
SOURCE	Creates, views, and manipulates source listing files.	See "SOURCE - Working with Source Listing Files" on page 3-10
AAREPORT	Creates, views, and manipulates Abend-AID report files.	See "AAREPORT - Working with Abend-AID Report Files" on page 3-37
CONTACTS	Associates contact information (for example, the phone number and e-mail address of an off-duty systems programmer) with a particular job name. When an abend occurs, users can display the contact information list, scan for a particular job name, and display the corresponding contact information.	See "Working with Contact Information" on page 8-9
LP	Displays an introduction to the Compuware Language Processor.	N/A
DDIO	Displays an overview of source listing and Abend-AID report files.	N/A
LEGAL	Displays the full Compuware copyright and trade secret notice.	N/A

Compuware/VF Entry Panel

Selecting VIEW on the Abend-AID primary menu displays the Compuware/VF Entry Panel. If you are an Abend-AID, Abend-AID for CICS, XPEDITER/TSO, or XPEDITER/CICS user, the screen in Figure 8-2 is displayed.

Here you specify the datasets you want to use. For datasets on remote MVS images, you also specify the Abend-AID server identifications that you need for Distributed Viewing Support (which requires that Base Services and HCI be installed). You can also set processing options.

Figure 8-2. Compuware/VF Entry Panel

```
COMPUWARE/VF ENTRY PANEL --------- COPYRIGHT COMPUWARE CORPORATION 1976, 2002.
Command ===>
Dataset choice . 0        (Select 0 - 8, from below, or C for Contact Data)
Member . . . . . ________ (MVS:jobname, SOURCE:program, or member*)
                          (Use the LNAME command to enter a Long Program Name)
                                                                   Remote
       Abend-AID Shared Directory or Report File:            Volume  Server
           0 'AA.REPORTS'
       Source Directory or Listing Files:
           1 'AA.LISTING'
           2
           3
           4
           5
           6
           7
           8
Processing options:                                 Source browsing options:
  Confirm delete  . .  YES      (Yes or no)           Unit . . . . . VIO
  Show print setup  .  YES      (Yes or no)           Blocking . . . 10
  Show source warning  YES      (Yes or no, used only for Abend-AID)
  Language  . . . . .  ENGLISH
  Local Server SSID .           (Required for Viewing Remote DDIO files)
```

Entry Panel Fields

The following sections discuss possible ways to complete each field on the Entry Panel. Your input is saved for future sessions.

Command Prompt

Enter one of the following:

- Valid TSO command
- CSSUTIL command to invoke CSS Utilities
- LNAME command to enter a long program name or long wildcard names to be used for member search. See Member description on page 8-4.

Note: See **Chapter 3, "CSS Utilities"** for a complete description of CSS Utilities.

Dataset Choice

Enter the number corresponding to a dataset listed in the Abend-AID Shared Directory or Report File, or Source Directory or Listing Files section of this screen. Or enter C for contact information.

Enter one of the following based on the type of output you desire:

- To view merged report and source information, enter 0 in the Dataset choice field.
- To view only a Source Listing File, enter 1 through 8 in the Dataset choice field.
- To view only a Source Listing File that is not currently available in the list, enter a number that is not currently being used (for example, in Figure 8-2 numbers 2 through 8 are not used). When you enter a number not currently being used, you must also enter the dataset name to the right of the corresponding number in the list.

Notes:

1. If a dataset number is not specified, 0 will be used.

2. When you press Enter after completing the remaining fields on the Compuware/VF Entry Panel, the Dataset Directory screen is displayed. For a description of this screen, see "Dataset Directory Screen" on page 8-10.

– To display contact information, enter C in the Dataset choice field.

The Contact Information feature allows you to associate contact information (for example, the phone number and e-mail address of the off-duty systems programmer) with a particular job name. If a job abends, users may search the contact information list to determine who to inform about the abend. For instructions on adding, editing, and viewing contact information, see "Working with Contact Information" on page 8-9.

Member

Specify the member name or group of member names of the file selected via your entry in the Dataset Choice field. Enter one of the following:

blank	Lists all members of the dataset choice above.
job name or program name	Lists all members in the file associated with the job name (for Abend-AID reports) or program name (for source listings). If the programname is in mixed case, you must enter it exactly as is or none of those members will be listed.
Member*	Lists all members in the file with the job or program names beginning with the characters specified before the wildcard character '*'. If the programname contains mixed case before the wildcard character, only the members that match that segment exactly will be listed. To enter a program name or wildcard longer than eight characters, you must use the LNAME command instead. The LNAME command is described in "Command Prompt" on page 8-3.

Abend-AID Shared Directory or Report File

To work with Abend-AID reports, enter the name of the Abend-AID report shared directory, report database, or report file. The dataset entered in this field (corresponding to Dataset choice 0) is the primary dataset. If the high-level qualifier of the dataset name is your TSO ID (prefix), begin with the next qualifier. Otherwise, enclose the dataset name in single quotation marks.

Source Directory or Listing Files

Enter the name of one or more source listing file(s) containing the output of the Compuware COBOL, PL/I, Assembler, or C language processor. If the high-level qualifier of the dataset name(s) is your TSO ID (prefix), begin with the next qualifier. Otherwise, enclose the dataset name(s) in single quotation marks.

This area lists up to eight source directory or listing files on lines 1 - 8. To view one of the listed source files, the number appearing to the left of the source file name (the line number) must be entered in the Dataset choice field.

The number on the line determines the order in which the datasets are searched. Blank lines can occur anywhere in the list.

Source Listing Database and Shared Directory source files can be entered in this list also.

Volume

If the DDIO file is not cataloged, or the catalog is not available, enter the serial number of the DASD volume containing the DDIO file.

Remote Server

If a dataset is located on a remote MVS system, identify the Abend-AID server on that system. Both the local and remote Abend-AID servers must be configured to use Base Services and HCI, which is required to support Distributed Viewing Support.

Processing Options

Use these options to specify the format of the output, and to display or suppress certain information.

Confirm delete

Specify whether to display a Confirm Delete screen before completing a delete request for the DDIO file. Valid entries are:

- **YES** - (Default) Display a Confirm Delete screen before completing a delete request.
- **NO** - Complete a delete request without displaying a Confirm Delete screen.

Show print setup

For hardcopy print requests, the print options determine the line size, page size, SYSOUT class, destination, route, and the maximum number of pages to be printed. You can choose to display a screen listing these options prior to printing a DDIO file's members allowing you to change the options as needed. Valid entries for this field are:

- **YES** - (Default) Display the print options prior to printing.
- **NO** - Do not display the print options. The currently defined options will be used.

Refer to "Hardcopy Options Screen" on page 8-19 for additional information.

Show source warning

Specify whether to display warning screens when the compile date, time, and language of the program in the Abend-AID report does not match a source listing in the source listing file. Valid entries are:

- **YES** - (Default) Display warning screens.
- **NO** - Do not display warning screens.

Note: If this field is set to **NO** and you have a mismatched source, the most recent available source listing will be selected. If this field is set to **NO,** and the source listing is not found, the basic Abend-AID report will be displayed.

Language

Specify the language and letter case (either uppercase or mixed-case) that you want to use for the Abend-AID report text. Valid entries are:

ENGLISH - (Default) Mixed-case English

JAPANESE - Japanese language support for terminals that support DBCS

USAMIX - Mixed-case English (same as ENGLISH)

USAUC - Uppercase English (select if your terminal does not support mixed case)

Note: If you will use the hardcopy facility to print Abend-AID reports, see "Line Commands" on page 8-18.

Local Server SSID

When using Distributed Viewing Support (which requires Base Services and HCI to be installed), this field is displayed for you to identify the server on the local MVS system. Both the local and remote Abend-AID servers must be configured for Distributed Viewing Support. Server IDs are assigned when the servers are configured for Distributed Viewing Support. This field is required to view a dataset on a remote server.

Source Browsing Options

These options create a temporary file that is used when browsing a source listing. The browse options use your system resources more efficiently.

Unit

Enter the type of device where the temporary file will be allocated when browsing a source listing. It can be any valid physical address, device type, generic name, or VIO. For example, use 2BF, 3380, SYSDA, or VIO. Recommended entries are VIO (if site-supported) or SYSDA.

Blocking

Enter the number of records in each block of the temporary file. A larger number may improve response time. If the number is too large, then the blocking factor is automatically changed to the maximum allowed for the specified device.

If you want Abend-AID to calculate the most efficient blocking for you, enter 9999. Abend-AID sets the blocking after you view a source listing.

Alternate Logon

If you entered specifications at the Entry Panel for access to a remote MVS system, the Compuware/VF displays the Alternate Logon screen. Here you log on to the Abend-AID server on that system.

Figure 8-3. Alternate Logon

```
------------------------- COMPUWARE/VF ALTERNATE LOGON -----------------------
 Command ===>




  Logon attempt to AASRV01  requires a USERID/PASSWORD. Please
  supply an alternate userid/password to complete access requirement
  for DDIO file: AA.R900.CXR706.RPTFILE


   Alternate USERID: REMOTEID    <== USERID on remote system
           PASSWORD:
```

Alternate USERID, PASSWORD

Enter your ID and password to access the remote Abend-AID server.

Viewing the Release Level

You can view the CSS release level in use at your site from the Compuware Viewing Facility Tutorial. Press the PF1 key to access the Tutorial. Type “About” in the Command field. The screen displays the CSS release level in use.

Parameters Required for Viewing Different Types of DDIO Files

The parameters required for viewing DDIO files will vary according to the type of DDIO file. Type the report shared directory file name in the Abend-AID Shared Directory or Report File field, if applicable. Type the source directory or listing file name(s) in the Source Directory or Listing Files field, if applicable.

To view Abend-AID Shared Directory or Report File with or without source support:

- Type **0** or a blank character in the Dataset choice field.

To view source listings:

- In the Dataset choice field, type a number (1 - 8).
- Specify the Source browsing options.

Creating JCL for LP Preprocessing and Postprocessing

Selecting CREATEJCL on the Abend-AID primary menu displays the CSS Utilities - CREATEJCL Menu (Figure 8-4). Options on this menu enable you to create and analyze a compiler listing and store the results of the analysis in a source listing DDIO file.

Figure 8-4. CSS Utilities - CREATEJCL Menu

```
----------- COMPUWARE CSS UTILITIES ---- Convert JCL Menu    -----      V8.0.0
OPTION ===>


       1  Preprocessor         - Create LP preprocessor JCL
       2  Postprocessor        - Create LP postprocessor JCL


      LP  Learn about Compuware language Processor
      SE  General Settings
      DD  Learn about Compuware DDIO files
      SD  Learn about Compuware Shared Directory and Database files









                 Enter END command to terminate
```

The CREATEJCL menu options are discussed in detail in **Chapter 3, "CSS Utilities"**. Please see "CREATEJCL - Preparing Compiler Listings for the LP" on page 3-4 for a complete description of these features.

Creating and Formatting Source Listing Files

Selecting SOURCE on the Abend-AID primary menu displays the CSS Utilities - Source Listing File Facility screen (Figure 8-5). Options on this screen enable you to create new source listing files, manipulate existing source listing files (for example: COPY, MOVE, EXPORT, IMPORT members), display file information, and delete files.

Figure 8-5. CSS Utilities - Source Listing File Facility

```
----------  CSS UTILITIES - Source Listing File Facility  -------- CSS V8.0.0
COMMAND ===>
Commands: SEtup   DDIOCALC (DDIO calculator)
                  DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

Processing Options for S/Member List:        Source Browsing Options:
    Confirm Delete ===> YES (Yes or No)               Unit ===> VIO
  Show print setup ===> YES (Yes or No)           Blocking ===> 10
          Language ===> ENGLISH           (ENGLISH/JAPANESE/USAMIX/USAUC)

SEL Options:  C (Create)  I (Info)  M (Move)  S (Member List)  A (Adv Function)

    SEL     --------------- Source Listing Files -------------------------

     _      ______________________________________________________________
     _  (1)  'CX.DDIO.LISTFILE'
     _  (2)  'CX.SOURCE.SHRDIR'
     _  (3)  'CX.SOURCE.DATABASE'
     _  (4)  'CX.SOURCE.DB1'
     _  (5)  'CX.R070601.DDIO'
     _  (6)
     _  (7)
     _  (8)
```

The Source Listing File Facility is discussed in detail in **Chapter 3, "CSS Utilities"**. See "SOURCE - Working with Source Listing Files" on page 3-10 for a complete description of these features.

Creating and Formatting Abend-AID Report Files

Selecting AAREPORT on the Abend-AID Primary Menu (Figure 8-1 on page 8-1) displays the CSS Utilities - Abend-AID Report File Facility (Figure 8-6). Options on this screen enable you to create new Abend-AID report files, manipulate existing Abend-AID report files (for example: COPY, MOVE, EXPORT, IMPORT members), display file information, and delete files.

Figure 8-6. CSS Utilities - Abend-AID Report File Facility

```
--------------  CSS UTILITIES - Abend-AID Report File Facility  ----- V8.0.0
COMMAND ===>
Commands: SEtup   DDIOCALC (DDIO calculator)
                  DD (DDIO tutorial)   SD (SD/DB tutorial)   LP (LP tutorial)

Processing Options for S/Member List:
    Confirm Delete ===> YES (Yes or No)
  Show print setup ===> YES (Yes or No)
          Language ===> ENGLISH           (ENGLISH/JAPANESE/USAMIX/USAUC)

SEL Options:  C (Create)  I (Info)  M (Move)  S (Member List)  A (Adv Function)

    SEL     --------------Abend-AID Report File(s)------------------------

     _      ______________________________________________________________
     _  (1)  'CX.TECH.TEST.DB80'
     _  (2)  'CX.REPTFILE'
     _  (3)  'CX.TECH.TEST.SDIR'
     _  (4)  'CX.ABENDAID.DATABASE'
     _  (5)  'CX.ABENDAID.SHRDIR'
     _  (6)  'CX.CREATE.NEW.FILE'
     _  (7)  'CX.LP.SRCEDIR'
     _  (8)
```

The Abend-AID report file facility is discussed in detail in **Chapter 3, "CSS Utilities"**. Please see "AAREPORT - Working with Abend-AID Report Files" on page 3-37 for a complete description of these features.

Working with Contact Information

The Compuware/VF contact information feature enables you to associate contact information (for example, the phone number and e-mail address of an off-duty systems programmer) with a particular job name. When an abend occurs, users can display the contact information list, scan for a particular job name, and display the corresponding contact information.

Note: If the contact information dataset has not been created by the systems programmer at your site, you will receive the error message "REGISTRATION FAILURE" when you attempt to use contact information. Refer to the *Enterprise Common Components Installation and Customization Guide*, Step 6 in the "CSS Customization" chapter for instructions on how to associate a Contact Information file with the Compuware/VF.

Maintaining Contact Data

To create, edit, and delete contact information, enter **C** at the Compuware/VF Entry Panel (Figure 8-7). The contact information file appears displaying all current contact entries.

Figure 8-7. Contact Information Entries

```
COMPUWARE/VF ------------'CX.CINFO' -------------------------- Row 1 to 2 of 2
COMMAND INPUT ===>                                              SCROLL ===> PAGE

C - View Contact Information   D - Delete   I - Insert   U - Update

  JOBNAME  ABEND CODE  DESC        Contact

  AA700MXT SOC4        PHILIPS     Duty Programmer
  CWX0101F SOC1        BROWN       Maintenance Staff
****************************** Bottom of data ********************************
```

The following line commands are available on this screen:

Table 8-2. Line Commands on Contact Information Screen

Enter	To
C	View the contact entry
D	Delete the contact entry
I	Insert (add) a new contact entry
U	Update a contact entry

If you enter the **I** or **U** line commands, a series of fields appear for you to complete or edit (Figure 8-8). You must complete the Jobname, Abend Code, and Desc fields. The remaining fields are optional. You may enter an asterisk (*) in the required fields to denote a wildcard.

Figure 8-8. Inserting or Updating a Contact Entry

```
COMPUWARE/VF ------------'CX.CINFO' ----------------------------------------
COMMAND INPUT ===>

 Update the following contact information:

*Jobname................
*Abend Code.............
*Desc...................
 Contact Name...........
 Phone Number...........
 Email Address..........
 Title..................
 Department.............
 Location...............
 Back-up Name...........
 Back-up Phone Number....
 Back-up Email Address...
 Special Instructions....

* Indicates required field.
```

Displaying Contact Data

When an abend occurs, users can view the contact information for a particular Abend-AID report by entering the **C** line command on the Abend-AID Dataset Directory screen (Figure 8-9). Note that the contact entry is view-only when displayed in this manner. To edit the contact information, see "Maintaining Contact Data" on page 8-9.

Dataset Directory Screen

The Dataset Directory screen displays the list of specified members in the primary DDIO file. By default, the members are listed in date and time order. Use the SORT primary command to alter the default order. Use the LOCATE and NEXT primary commands and the standard ISPF/PDF scrolling commands and PF keys to locate a member. You can also use the VIEW command to control the display of Assembler source lines in an Abend-AID report that is merging Assembler source with Abend-AID text.

The layout of the Dataset Directory screen depends on the type of DDIO file. Each file type is discussed in the following sections.

Abend-AID Dataset Directory

The Abend-AID Dataset Directory screen (Figure 8-9) displays the list of members in the file specified in the Abend-AID Report File field of the Compuware/VF Entry Panel (Figure 8-2 on page 8-3). By default, the members are listed in order according to date and time on the Compuware/VF Entry Panel.

Figure 8-9. Abend-AID Dataset Directory Screen

```
COMPUWARE/VF ----- 'COMPUWARE.REPORT.FILE' ------------------- Row 1 to 4 of 4
COMMAND INPUT ===>                                            SCROLL ===> PAGE
                      Abend-AID DATASET DIRECTORY

      S - View C - Contact Info D - Delete P - Print L - Lock U - Unlock

   JOBNAME     REPT NUMBER  JESID CODE      DATE       TIME   DESC      SIZE(K)
  AA700MXT (L)         273 J26165 S0CB   13 OCT 2001  8.29.14 PHILIPS       160
  AA700MXT             272 J20423 S0CB   12 OCT 2001  8.27.30 SMITH         160
  CWX0101F (M)         270 J04608 S0CB   30 SEP 2001 15.03.20 BROWN         639
  EFHTPT0A             269 J26107 S0CB   28 SEP 2001 16.35.08 JOHNSON     10871
****************************** Bottom of data ******************************
```

Figure 8-10. Abend-AID Dataset Directory Screen for Shared Directory

```
COMPUWARE/VF ----- 'CX.NEW.AA.SD' ---------------------------- Row 1 to 2 of 2
COMMAND INPUT ===>                                            SCROLL ===> CSR
                Abend-AID DATASET DIRECTORY - Shared Directory

      S - View C - Contact Info D - Delete P - Print L - Lock U - Unlock

   JOBNAME   REPT NUMBER  JESID  CODE       DATE       TIME   DESC      SIZE(K)
  AA944THD             9 J31411  S0C1    01 FEB 2002  7.52.53 AA 9.4 SD       0
  AA94COBD (M)         1 J28254  S0C7    21 FEB 2002 13.45.24 Programmer      0
****************************** Bottom of data ********************************
```

The Abend-AID Dataset Directory Screens (Figure 8-9 and Figure 8-10) display the following information:

JOBNAME

The name of the abending job.

REPT NUMBER

The number assigned by Abend-AID to identify the diagnostic report. The Abend-AID reports are numbered sequentially, beginning with 1 and continuing until the report file is reformatted. The following status indicators also appear, if applicable, in the **Rept Number** column.

- ***I** Abend-AID report is incomplete.
- ***IL** Abend-AID report is incomplete and automatically locked.
- ***IM** Abend-AID report is incomplete and manually locked.
- **(L)** Abend-AID report is automatically locked.
- **(M)** Abend-AID report is manually locked.

JESID

The job number assigned by the Job Entry Subsystem.

CODE

The system or user abnormal termination code, or “SNAP” for a SNAP-AID report, or PL/I on-code.

DATE

The date on which the error occurred or the date of the SNAP-AID report.

TIME

The time at which the error occurred, or the time the SNAP-AID report was created.

DESC

By default, the first ten characters of the **Programmer Name** field from the job card statement. When using Abend-AID exit CWEXIT02, or when calling SNAP-AID, you can override this value, in which case the new value will be shown. See the Abend-AID documentation for more information.

SIZE

The number of bytes (KB) allocated for the Abend-AID report in the report file.

Figure 8-11 shows the Abend-AID Dataset Directory screen for Shared Directory after the “SHOWDB” command was entered. This command will show which attached database the member resides on and functions only on this screen. Enter **SHOWDB OFF** to turn off the database name display.

Figure 8-11. Abend-AID Dataset Directory Screen for Shared Directory after SHOWDB Command

```
COMPUWARE/VF ----- 'CX.NEW.AA.SD' ---------------------------- Row 1 to 2 of 2
COMMAND INPUT ===>                                            SCROLL ===> CSR
                Abend-AID DATASET DIRECTORY - Shared Directory

      S - View C - Contact Info D - Delete P - Print L - Lock U - Unlock

   JOBNAME   REPT NUMBER  JESID  CODE       DATE       TIME   DESC      SIZE(K)
  AA944THD             9 J31411  S0C1    01 FEB 2002  7.52.53 AA 9.4 SD      80
  ABENDAID DATABASE:  CSS.QA.AADB2
  AA94COBD (M)         1 J28254  S0C7    21 FEB 2002  3.45.24 Programmer     80
  ABENDAID DATABASE:  CSS.QA.AADB1
******************************** Bottom of data ********************************
```

Source Listing Dataset Directory

The Source Listing Dataset Directory (Figure 8-12 and Figure 8-13) displays all source listings in a standard DDIO Member List or Shared Directory Member List.

Figure 8-12. Source Listing Standard DDIO Member List Screen

```
COMPUWARE/VF -- Source Listing File Member List -----         ROW 1 TO 2 OF 2
COMMAND INPUT ===>                                            SCROLL ===> CSR



SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
______ FEA02P2               2    05 JUN 2002 08.49.11  04   PL/I V1R5     120
______ fea02p2               1    05 JUN 2002 08.49.11  04   PL/I V1R5     120
******************************** Bottom of Data ********************************
```

Note: Selection **I-Info** is valid only for Source Listing Database and Shared Directory.

Figure 8-13 shows the Source Listing Shared Directory Member List screen with “display long PROGRAM names” set to NO. A LONGNAME will be indicated by an ‘*’ in the eighth position of the name. A mixed case name is shown as it is specified in the program or by the PROGRAM-NAME parameter.

Figure 8-13. Source Listing Shared Directory Member List Screen (Not LONGNAME)

```
COMPUWARE/VF -- Source Listing Shared Directory Member List -- Row 1 to 7 of 7
COMMAND INPUT ===>                                            SCROLL ===> CSR
Shared Directory: CX.LP.SHRDIR

Display Long PROGRAM Names ==> NO  (yes or no) Number of lines ==>  1  (1 TO 4)

SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock   I - Info
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
______ AVery     (L)       215    07 SEP 2001  08.55.34  00   C/390 V2R6   1032
______ ASMTEST   (L)       199    21 APR 2000  15.42     00   ASMH           60
______ COBABND6  (M)       192    18 APR 2000  06:47:02  00   COBIIR4        48
______ AVeryLa*  (L)       191    14 APR 2000  11:16:29  00   C/ESA V3R2    816
______ COBABND1  (M)       189    12 APR 2000  12:38:15  04   COBIIR4        36
______ COBABEND  (M)       186    04 APR 2000  17:11:10  04   COBIIR4        36
______ COBABND1              7    04 APR 2000  17:11:31  04   COBIIR4        36
******************************** Bottom of data ********************************
```

Figure 8-14 shows the Source Listing Shared Directory Member List after the “SHOWDB” command was entered. This command is to show which attached database the member resides on and only functions on this screen. Enter **SHOWDB OFF** to turn off the database name display.

Figure 8-14. Source Listing Shared Directory Member List Screen after SHOWDB Command (Not LONGNAME)

```
COMPUWARE/VF -- Source Listing Shared Directory Member List  -- Row 1 to 5 of 7
COMMAND INPUT ===>                                            SCROLL ===>CSR
Shared Directory: CX.LP.SHRDIR

Display Long PROGRAM Names ==> NO  (yes or no)  Number of lines ==>  1 (1 TO 4)

SELECT S - Select   P - Print   D - Delete   L - Lock    U - Unlock   I - Info
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME     RC   LANGUAGE  SIZE(K)
______ AVery     (L)       215    07 SEP 2001 08.55.34    00   C/390 V2R6   1032
SOURCE DATABASE: CX.LP.DATABASE
______ ASMTEST   (L)       199    21 APR 2000 15.42       00   ASMH           60
SOURCE DATABASE: CX.LP.DATABASE
______ COBABND6  (M)       192    18 APR 2000 06:47:02    00   COBIIR4        48
SOURCE DATABASE: CX.LP.DATABASE
______ AVeryLa*  (L)       191    14 APR 2000 11.16.29    00   C/ESA V3R2    816
SOURCE DATABASE: CX.LP.DATABASE
______COBABND    (M)       189    12 APR 2000 12:38:15    04   COBIIR4        36
SOURCE DATABASE: CX.LP.DATABASE
****************************** Bottom of data ********************************
```

Figure 8-15 shows the Source Listing Shared Directory Member List with “display long PROGRAM names” set to YES. A mixed case name is shown as it is specified in the program or by the PROGRAM-NAME parameter.

Figure 8-15. Source Listing Shared Directory Member List Screen (LONGNAME)

```
COMPUWARE/VF -- Source Listing Shared Directory Member List -- Row 1 to 5 of 7
COMMAND INPUT ===>                                            SCROLL ===> CSR
Shared Directory: CX.LP.SHRDIR

Display Long PROGRAM Names ==> YES (yes or no) Number of lines ==>  1  (1 TO 4)

SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock   I - Info
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
______ AVery     (L)       215    07 SEP 2001  08.55.34  00   C/390 V2R6   1032
AVery
SOURCE DATABASE: CX.LP.DATABASE
______ ASMTEST   (L)       199    21 APR 2000  15.42     00   ASMH           60
ASMTEST
SOURCE DATABASE: CX.LP.DATABASE
______ COBABND6  (M)       192    18 APR 2000  06:47:02  00   COBIIR4        48
COBABND6
SOURCE DATABASE: CX.LP.DATABASE
______ AVeryLa*  (L)       191    14 APR 2000  11:16:29  00   C/ESA V3R2    816
AVeryLargeNameofatLeast30Characters_20
SOURCE DATABASE: CX.LP.DATABASE
______ COBABND1  (M)       189    12 APR 2000  12:38:15  04   COBIIR4        36
COBABND1
SOURCE DATABASE: CX.LP.DATABASE
```

Figure 8-16 shows the Source Listing Database Member List screen with “display long PROGRAM names” set to NO. A LONGNAME will be indicated by an ‘*’ in the eighth position of the name. A mixed case name is shown as it is specified it is specified in the program or by the PROGRAM-NAME parameter.

Figure 8-16. Source Listing Database Member List Screen (Not LONGNAME)

```
COMPUWARE/VF -- Source Listing Database Member List -----     Row 1 to 9 of 9
COMMAND INPUT ===>                                            SCROLL ===> CSR
Source Database:  CX.LP.DATABASE
ATTACHED TO:      CX.LP.SHRDIR
Display Long PROGRAM Names ==> NO  (yes or no) Number of lines ==>  1  (1 TO 4)

SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock   I - Info
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
______ ASMTEST             415    26 OCT 2001  15.12     00   ASMH           56
______ COBDEMO1            405    03 OCT 2001  15:41:41  04   COBIIR4        98
______ COBDEMO1            404    27 SEP 2001  13:23:08  04   COBIIR4        98
______ COBDEMO1            403    20 SEP 2001  09:52:21  04   COBIIR4        98
______ COBDEMO1  (L)       402    10 SEP 2001  16:03:47  04   COBIIR4        98
______ ASMTEST   (L)       400    10 SEP 2001  15.55     00   ASMH           56
______ ASMTEST             398    10 SEP 2001  15.08     00   ASMH           56
______ AVeryBi*  (M)       397    06 SEP 2001  15.16.50  00   C/390 V2R6   1035
______ AVeryBi*  (M)       375    21 FEB 2000  13.06.52  00   C/ESA V3R2    826
****************************** Bottom of data ******************************
```

Figure 8-17 shows the Source Listing Database Member List screen with “display long PROGRAM names” set to YES. A mixed case name is shown as it is specified in the program or by the PROGRAM-NAME parameter.

Figure 8-17. Source Listing Database Member List Screen (LONGNAME)

```
COMPUWARE/VF -- Source Listing  Database Member List -----    Row 5 to 9 of 9
COMMAND INPUT ===>                                            SCROLL ===> CSR
Source Database:  CX.LP.DATABASE
ATTACHED TO:      CX.LP.SHRDIR
Display Long PROGRAM Names ==> YES (yes or no) Number of lines ==>  1  (1 TO 4)

SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock   I - Info
CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
______ COBDEMO1  (L)       402    10 SEP 2001  16:03:47  04   COBIIR4        98
COBDEMO1
______ ASMTEST   (L)       400    10 SEP 2001  15.55     00   ASMH           56
ASMTEST
______ ASMTEST             398    10 SEP 2001  15.08     00   ASMH           56
ASMTEST
______ AVeryBi*  (M)       397    06 SEP 2001  15.16.50  00   C/390 V2R6   1035
AVeryBigLongNameofatLeast30Characters_20
______ AVeryBi*  (M)       375    21 FEB 2000  13.06.52  00   C/ESA V3R2    826
AVeryBigLargeNameofatLeast30Characters_20
****************************** Bottom of data ******************************
```

The Source Listing Dataset Directory Screen displays the following information.

Program

The member names (source listing names) in the source listings file. For Source Listing Database and Source Listing Shared Directory files, a name longer than eight characters will be displayed with an ‘*’ in the eighth position. Use either an I line command or YES on “Display Long Program Name” to see the full LONGNAME.

LIST NUMBER

The number assigned by the language processor to identify the source listing. Source listings are numbered sequentially, beginning at 1 and continuing until the source listing file is reformatted. The following status indicators also appear, if applicable, in the **List Number** column.

(L) Source listing is automatically locked.

(M) Source listing is manually locked using the L line command or the LOCK batch command.

***I** Source listing is incomplete.

***IL** Source listing is incomplete and automatically locked.

***IM** Source listing is incomplete and manually locked.

COMP DATE

The date the program was compiled.

TIME

The time the program was compiled.

RC

The Compuware language processor return code.

LANGUAGE

The program language compiler and version. The following codes can be found in the LANGUAGE column:

ASMF	Assembler F
ASMH	Assembler H
HLASM	High Level Assembler 1.1
HLASM R2	High Level Assembler 1.2
HLASM R3	High Level Assembler 1.3
HLASM R4	High Level Assembler 1.4
OSVSCOB	OS/VS COBOL
OSVSCOBOPT	OS/VS COBOL/CA-Optimizer
C/ESA V3R2	“C” for ESA V3R2
C/390 V1R2	OS/390 “C” V1R2
C/390 V1R3	OS/390 “C” V1R3
C/390 V2R4	OS/390 “C” V2R4
C/390 V2R5	OS/390 “C” V2R5
C/390 V2R6	OS/390 “C” V2R6
C/390 V2R9	OS/390 “C” V2R9
C/390 V2RA	OS/390 “C” V2R10
COBIIR2	VS COBOL II Release 2
COBIIR2OPT	VS COBOL II Release 2/CA-Optimizer II
COBIIR3	VS COBOL II Release 3
COBIIR3OPT	VS COBOL II Release 3/CA-Optimizer II
COBIIR4	VS COBOL II Release 4
COBIIR4 JA	VS COBOL II Release 4 with Japanese Language Support
COBIIR4OPT	VS COBOL II Release 4/CA-Optimizer II
COBII JA	VS COBOL II with Japanese Language Support
COBOL/370	IBM SAA AD/Cycle COBOL/370
COBOL/370 JA	IBM SAA AD/Cycle COBOL/370 with Japanese Language Support
COBOL/370OPT	IBM SAA AD/Cycle COBOL/370/CA-Optimizer II
COBOL/390	COBOL for OS/390 and VM
COBOL/390M	COBOL for OS/390 and VM with MLE (Millennium Language Extensions)

COBOL/MVSM	COBOL for MVS and VM with MLE
COB/370JA	COBOL/370 Japanese Language Support
COB/390OPM	COBOL/390 MLE CA-Optimizer
COB/390OPT	COBOL/390 CA-Optimizer
COB/MVS	COBOL for MVS and VM
COB/MVS JA	COBOL for MVS and VM with Japanese Language Support
COB/MVSOPM	COBOL/MVS MLE CA-Optimizer
COB/MVSOPT	COBOL/MVS CA-Optimizer
COBOL Z/OS	Enterprise COBOL for z/OS and OS/390.
COBOL Z/OSJA	Enterprise COBOL for z/OS and OS/390 with Japanese language support.
COB/Z V3R2	Enterprise COBOL for z/OS and OS/390 Release 3.2
COB Z/OSJA	Enterprise COBOL for z/OS and OS/390 Release 3.2 with Japanese Language Support
PL/I V1R5	PL/I V1
PL/I V2	PL/I V2
PL/I AD V1	PL/I for MVS and VM
PL1VA V2R2	PL/I VisualAge V2R2
PL1EN V3R1	Enterprise PL/I for z/OS
PL1EN V3R2	Enterprise PL/I for z/OS

SIZE

The number of bytes (KB) allocated for the source listing in the source listing file.

Note: If you are viewing an Assembler-compiled program and the Viewing Facility cannot find a unique match for an Assembler source listing (required for merging text and source), the Assembler Mismatch List screen (Figure 8-31) is displayed.

Remember, when using multi-CSECT Assembler programs, the CSECT you are looking for may be in a source listing of an entirely different name. Type an **s** next to the correct source listing and press Enter, or press PF3 to view the report without a source listing.

For information about Assembler mismatched structure, see “Mismatched Structure for Assembler” on page 8-33.

Primary Commands

In addition to the standard scrolling commands, you can enter the following commands on the COMMAND INPUT area on the Dataset Directory screens (Figure 8-9 on page 8-10 and Figure 8-12 on page 8-12):

LOCATE (L) jobname/programname

Positions the matched **jobname** (Abend-AID report files) or **programname** (source listing files) as the first entry on the Abend-AID Dataset Directory screen.

The following diagram illustrates the syntax for the LOCATE command.

```
►►─Locate──string──────────►◄
```

string

Any character string from one to eight characters in length. The asterisk (*) wildcard character is valid at the end of the string when the string is less than eight characters long.

Note: String is case sensitive and must match the case of the program or job name.

NEXT (N)

Scrolls forward to the next JOBNAME (Abend-AID report files) or PROGRAM NAME (source listing files) specified in the previous LOCATE command. You must have first issued a LOCATE command before you can issue the NEXT command.

The following diagram illustrates the syntax for the NEXT command.

```
                 ┌1──────┐
►►─Next──────────┼───────┼──────────►◄
                 └number─┘
```

number

Specifies the next Nth match to be located. The search starts at the second displayed line, and continues toward the end of the list until N matches are made. If there are less than N matches, the last found line will be scrolled to the top of the screen. If there are no matches, the screen remains unchanged.

Valid entries are any number from 1 to 9999. The default value is 1.

SORT

Sorts the member list by any field in the directory except TIME. The following diagram illustrates the syntax for the SORT command.

```
►►─SORT──────sort-key──────►◄
```

sort-key

The following tables list the valid sort keys, their abbreviations, and the resulting sorting sequence for the different types of DDIO files:

Table 8-3. Abend-AID Report File Sort Keys

Sort Key	Abbreviation	Primary Sort Field	Sequence	Secondary Sort Field	Sequence	Tertiary Sort Field	Sequence
JOBNAME	JN	JOBNAME	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
JESID		JESID	Descending	DATE/TIME	Descending	REPT-NUMBER	Descending
DESCRIPTION	DESC	DESCRIPTION	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
SIZE		SIZE	Descending	DATE/TIME	Descending	REPT-NUMBER	Descending
NUMBER	NUM	REPT- NUMBER	Descending				
CODE		CODE	Ascending	DATE/TIME	Descending	REPT-NUMBER	Descending
DATE		DATE/TIME	Descending	REPT-NUMBER	Descending		

Table 8-4. Source Listing File Sort keys

Sort Key	Abbreviation	Primary Sort Field	Sequence	Secondary Sort Field	Sequence	Tertiary Sort Field	Sequence
PROGRAM	PGM	PROGRAM	Ascending	DATE/TIME	Descending	LIST-NUMBER	Descending
NUMBER	NUM	LIST-NUMBER	Descending				
DATE		DATE/TIME	Descending	LIST-NUMBER	Descending		
RC (return code)		RC	Ascending	DATE/TIME	Descending	LIST-NUMBER	Descending
LANGUAGE	LANG	LANGUAGE	Ascending	DATE/TIME	Descending	LIST-NUMBER	Descending
SIZE		SIZE	Descending	DATE/TIME	Descending	LIST-NUMBER	Descending

Line Commands

After a Dataset Directory Screen is displayed, the following line commands can be entered to the left of the directory entries.

D Deletes the member (a report or source listing) from the DDIO file. If you entered YES in the **CONFIRM DELETE f**ield of the Entry Panel, a Confirm Delete screen will be displayed before the member is deleted. If the member is locked (automatically or manually), the member is unlocked and then deleted.

L Manually locks the member. While the member is manually locked, it cannot be automatically deleted when the DDIO file becomes full. Manual locks display with a status indicator of **M** on the directory screen.

I Additional information showing full name (for LONGNAME) and the Source Listing Database and/or Source Listing Shared Directory name(s) and VOLSER(s). Only valid for Source Listing Database and Shared Directory files.

P Prints the member (report or source listing) using the hardcopy facility. If you entered YES in the **SHOW PRINT SETUP** field on the Entry Panel, the Hardcopy Options Screen will be displayed.

S Selects the member (report or source listing) to be viewed.

U Unlocks the member so it can be automatically deleted when the DDIO file formatted with AUTODELETE=YES, AUTODELETE=DUPS, or AUTODELETE=STAGED becomes full.

You can enter multiple line commands, which will be executed from top to bottom.

Figure 8-18. Source Listing Shared Directory Member List Screen (After Entering the Line Command “I” (Info))

```
COMPUWARE/VF -- Source Listing Shared Directory Member List -- Row 4 to 7 of 7
 COMMAND INPUT ===>                                            SCROLL ===> CSR
 Shared Directory: CX.LP.SHRDIR

 Display Long PROGRAM Names ==> NO  (yes or no) Number of lines ==>  1  (1 TO 4)

 SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock   I - Info
 CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
 ______ AVeryLa*  (L)       191                           00   C/ESA V3R2    816
 _ +-------------------------------------------------------------------------+ 36
 _ |------------------------- Program Information -------- Row 1 to 1 of 1  | 36
 _ |COMMAND INPUT ===>                                  SCROLL ===> CSR     | 36
 * |Database: CX.LP.DATABASE                              Volser: PRD964    | **
   |                                                                        |
   |                                                                        |
   | Full Program Name:                                                     |
   | AVeryLargeNameofatLeast30Characters_20                                 |
   | ************************** Bottom of data ***************************  |
   |                                                                        |
   |                                                                        |
   |                                                                        |
   |                                                                        |
   +------------------------------------------------------------------------+
```

Figure 8-19. Source Listing Database Directory Screen (After Entering the Line Command “I” (Info))

```
COMPUWARE/VF -- Source Listing  Database Member List -----     Row 9 to 9 of 9
 COMMAND INPUT ===>                                            SCROLL ===> CSR
 Source Database:  CX.LP.DATABASE
 ATTACHED TO:      CX.LP.SHRDIR
 Display Long PROGRAM Names ==> NO  (yes or no) Number of lines ==>  1  (1 TO 4)

 SELECT S - Select   P - Print   D - Delete   L - Lock   U - Unlock   I - Info
 CODE   PROGRAM     LIST NUMBER     COMP DATE     TIME    RC   LANGUAGE  SIZE(K)
 ______ AVeryBi*  (M)       375    21 FEB 2000  13.06.52  00   C/ESA V3R2    826
_ +-------------------------------------------------------------------------+ 36
_ |  ------------------------- Program Information -------- Row 1 to 1 of 1 | 32
_ |  COMMAND INPUT ===>                                  SCROLL ===> CSR    | 32
_ |  Database: CX.LP.DATABASE                              Volser: PRD910   | 36
_ |  Shared Directory: CX.LP.SHRDIR                        Volser: PRD988   | 32
_ |                                                                         | 32
_ |  Full Program Name:                                                     | 32
_ |  AVeryBigLargeNameofatLeast30Characters_20                              | 32
_ |  ************************** Bottom of data ***************************  | 32
_ |                                                                         | 28
_ |                                                                         | 32
_ |                                                                         | 32
_ |                                                                         | 88
_ |                                                                         | 36
_ +-------------------------------------------------------------------------+ 96
```

Hardcopy Options Screen

The Compuware/VF Hardcopy Options Screen (Figure 8-20), shown with default settings, enables you to set options for printing a copy of the DDIO file member. This screen is displayed if you entered the **P** line command on the Dataset Directory Screen and specified YES in the **SHOW PRINT SETUP** field on the Entry Panel.

Figure 8-20. Compuware/VF Hardcopy Options Screen

```
 COMPUWARE/VF HARDCOPY OPTIONS
 COMMAND INPUT ===>

 Current DDIO file: AA.REPORTS
            Member: SBL60LN3
Report/List Number: 118

Enter/verify options below:
   Line size           ===> NARROW                 (NARROW or WIDE)
   Page size           ===> 60                     (20 to 99)
   Sysout class        ===>                        (class,writer-name,format)
   Destination         ===>                        (nodename,userid)

   Route               ===> SYSOUT                 (SYSOUT or BATCH)
   Max number of pages ===>                        (for Route=BATCH)
   Unit                ===> SYSDA                  (for Route=BATCH)
```

CURRENT DDIO FILE

The file name from which you are printing a member.

MEMBER

The member you selected to print.

REPORT/LIST NUMBER:

The report or list number of the member you selected to print.

LINE SIZE

Specify the width of the printed copy of the DDIO file member. Valid entries are:

NARROW: (Default) Prints optional 80-column output.

WIDE: Prints standard 133-column output.

PAGE SIZE

Enter the number of lines per page for printed output. Valid entries are 20 through 99. The default value is 60.

SYSOUT CLASS

Specify the JES spool class where the output is printed. The JCL formats CLASS and (CLASS,WRITER-NAME,FORMAT) are acceptable.

DESTINATION

Enter the JES destination code where the output is routed. The JCL formats NODENAME and (NODENAME,USERID) are acceptable.

ROUTE

Specify how the report or source listing file member is to be printed. Enter one of the following:

- Use SYSOUT to allocate a SYSOUT dataset into which the member will be copied. Job information is obtained from the security profile attached to your TSO account. Your user ID is used as the JOBNAME.
- Use BATCH to allocate a work file (DSN beginning with “prefix.DDSUTL.”) into which the member is copied. If there is no TSO prefix, the DSN will begin with “userid.DDSUTL.”. A job is submitted to JES using the ISPF default job card from the ISPF 0.2 screen to print and then delete the file.

MAX NUMBER OF PAGES

Determine the file size of the work file when the BATCH route value is specified.

UNIT

Enter the device type for the work file when the BATCH route value is specified. It can be any valid physical address, device type, generic name, or VIO. (For example, use 2BF, 3380, SYSDA, or VIO).

Viewing Source Listings

To view a source listing file, select it using the **S** line command on the Source Listing Dataset Directory screen (Figure 8-12 on page 8-12). This allocates a temporary file, copies the source listing to this file, and calls the ISPF BROWSE function to display the source listing. Use the standard ISPF BROWSE commands to view the source listing.

Viewing Abend-AID Reports

Enter an S to the left of an Abend-AID report on the Abend-AID Dataset Directory Screen (Figure 8-9 on page 8-10) to display the Output Selection Menu (Figure 8-21 on page 8-22). This screen enables you to display specific sections of the Abend-AID report.

The menu lists all possible sections in an Abend-AID report. Some sections may not exist for a specific Abend-AID report if they are not applicable. If your report does not contain a particular section, and you attempt to select that section from the menu, you will receive a SECTION NOT FOUND message.

From within a selected section, you can view any other section (without returning to the Output Selection Menu) by entering the section number or name abbreviation in the **COMMAND INPUT** field.

Certain fields that are highlighted are designated as cursor-selectable point-and-shoot fields. However, these fields will only be highlighted if the corresponding source information was found. Position the cursor within the highlighted field and press Enter to transfer to the related source information. Press PF3 to return to the original display.

Stacking of cursor point-and-shoot panels is limited to eight. Trying to point-and-shoot more than eight times without any PF3/returns will result in an error message.

Viewtime Restriction

The display of PL/I uninitialized external static structure storage is restricted to structures having no alignment bytes at the beginning of the CSECT. In cases where PL/I has added bytes to the front of a static external structure, those bytes will be included in the display of the data. This causes the entire structure to be displayed in hexadecimal:

```
DCL 1 SARRAY1 EXT,
    2 NAME1 CHAR(11),
    2 BINA1 BIN FIXED(31);
```

Output Selection Menu for Abend-AID

After you select an Abend-AID report for viewing, the Output Selection Menu (Figure 8-21) is displayed.

Figure 8-21. Compuware/VF Output Selection Menu for Abend-AID

```
COMPUWARE/VF ---------------- OUTPUT SELECTION MENU ---------------------------
 SELECT SECTION ===>

 'AA.REPORT'  REPT.7  AA94CB16  J211074

  0 HEADER   System and job information
  1 DIAGS    Analysis
  2 NSI      Location (next sequential instruction)
  3 TRACE    Program flow and program attributes
  5 REGS     Register contents and PSWs
  6 PROG     Program Storage
  7 PLIST    Program Listing
  8 FILES    Open files information
  9 IMS      IMS information and PCBs
 10 IDMS     CA-IDMS information
 11 DB2      DB2 information
 12 SORT     COBOL sort information
 13 DATALOC  COBOL Data Locator
 14 MQ       Websphere MQ information
 15 LE       LE information
 20 EPILOG   Abend-AID termination information

                  Key section number or name and press ENTER.
     Or press ENTER to view the diagnostic summary of sections 1 through 5.
```

Use the Compuware/VF Output Selection Menu (Figure 8-21) to view specific sections of the Abend-AID report. The sections that are available for viewing are highlighted with reverse video and are cursor-selectable. In order to view a particular section, enter the section number or the abbreviation for the section name in the SELECT SECTION field or tab to the desired section and press **Enter**. To view sections 0 through 5, you can either enter a section number or name abbreviation to begin viewing at a specific section. The default if you press **Enter** is to begin viewing these sections from section 1 - DIAGS (Analysis of Error).

SCALE1

Displays vertical hex storage relative to one. (default)

SCALE0

Displays vertical hex storage relative to zero.

The above primary commands can be entered on the Output Selection Menu or any subsection's command line. This will display any vertical hex storage in the style that was specified. After the command is specified, it will be stored in the user's TSO profile and continued to be displayed in the manner requested until it is changed by the user.

Options 0-5

If you select any section number from 0 through 5, all existing sections with numbers 0 through 5 can be viewed. If you select any section number greater than 5, only the selected section can be viewed. You can use standard ISPF scroll commands to scroll within the viewable portion of the Abend-AID report.

0 or Header

Top of output, including the header information and available help information.

1 or DIAGS

Diagnostic information (Analysis of Errors). For COBOL, this section may highlight the statement number on the statement where the abend occurred. Selecting this cursor-selectable section will take you to that portion of the program listing. Also for COBOL, this section may highlight the associated field names. Selecting one of these field names will take you to that field in the working storage section in program storage.

2 or NSI

Instruction section (Error Location). This section may have two highlighted statement numbers — one under “error location” and the other under “last known I/O request or call”. Selecting either of these cursor-selectable fields will take you to that location in the program listing.

3 or TRACE

Call Trace Summary and Application Program Attributes. The call trace summary may have one or more highlighted offset values (under the “return/value” column). Selecting one of these cursor-selectable offset values will take you to that location in the program listing for that calling program.

5 or REGS

Register contents and PSW.

Option 6 — Program Storage

If you choose option 6 on the Compuware/VF Output Selection Menu (Figure 8-21 on page 8-22) and only one CSECT exists in the output, that Program section will be displayed. If more than one CSECT exists in the output, the Compuware/VF Program Storage Selection List (Figure 8-22) is displayed.

Figure 8-22. Compuware/VF Program Storage Selection List

```
COMPUWARE/VF  PROGRAM STORAGE SELECTION LIST -------------- ROW 1 TO 2 OF 2
COMMAND INPUT ===> _                                        SCROLL ===> CSR

  PROGRAM / PROCEDURE NAME      LOAD    LANGUAGE     COMPILE DATE  -  TIME

  PAYROLL2                    PAYROLL2  VS COBOL II  23 FEB 1996      13.34.53
  MONTHS2                     PAYROLL2  VS COBOL II  23 FEB 1996      13.34.32
****************************** BOTTOM OF DATA ******************************
```

This screen lists all PROGRAMs, CSECTs, or PL/I procedures in the Abend-AID report. The following information is displayed in the selection list:

PROGRAM

The CSECT or PL/I procedure name of the program.

LOAD

The name of the load module containing the program.

LANGUAGE

The language in which the program was written.

COMPILE DATE - TIME

For OS/VS COBOL, VS COBOL II, IBM SAA AD/Cycle COBOL/370, COBOL/390, and PL/I programs, the date and time when the program was compiled.

Selecting Program Storage

To select program storage for viewing, enter the line command **S** to the left of the program name.

When you want to select specific program storage:

- from the Output Selection Menu or within any Abend-AID report section, and
- when not displaying the Program Storage Selection List first, and
- when more than one CSECT exists in the output

enter the primary command:

- 'EXT' for OS/VS COBOL II or IBM SAA AD/Cycle COBOL/370 EXTERNAL DATA, or
- 'PROG(xxxxxxxx)' for program xxxxxxxx.

Viewing Program Storage Areas

Compuware/VF displays data areas as soon as they are formatted enabling you to view and search available data immediately while continuing to format the rest of program storage. A status message appears in the upper right corner of the screen indicating whether formatting is still in progress or if it has completed.

While formatting is in progress, you can use the PF7 and PF8 keys to scroll through the currently-formatted data areas. You can also use the FIND command to conduct searches in the currently-formatted text.

If the program name is highlighted, it indicates the field is cursor-selectable and, if selected, will transfer display to the associated program listing ("Option 7—Program Listing" on page 8-25).

For COBOL only, if the FD file name is highlighted, it indicates the field is cursor-selectable and, if selected, will transfer display to the associated data management control blocks ("Option 8—Data Management Control Blocks" on page 8-26).

Subsection Commands

COBOL

Subsection commands are available for COBOL Working Storage displays. They can be entered when viewing a selected program storage section and also from any Abend-AID report section in combination with the section command PROG(xxxxxxxx).

The available COBOL subsection commands are:

EXT	Displays the COBOL II External Data section
TGT	Position at the TGT.

Note: The following **BL** commands are available only when you are viewing the Abend-AID report with basic support. The SOURCE command must be set to OFF to use these commands.

BL yyy	Position at COBOL BL(W) cell **yyy**
BLL yyy	Position at COBOL BLL cell **yyy.**
BLW yyy	Position at COBOL BLW cell **yyy.**
BLX zzz	Position at COBOL BLX cell **zzz.**

where **yyy** can be nnn for the decimal cell number or X'nnn' for the hexadecimal cell number, and **zzz** can be X'nnn' for the hexadecimal address of the cell.

For example, BLL 25 will position the current program display at BLL cell 25, and PROG(MYPROG) BLW X'25' will display Working Storage for program MYPROG positioned at BLW cell 37 (X'25').

Note: If **yyy** or **zzz** is omitted, the output will be positioned at the first cell of the specified type.

PL/I

Subsection commands are available for the PL/I procedure displays. They can be entered when viewing a selected section and also from any Abend-AID report section

in combination with the section command PROG(xxxxxxxx). PL/I subsection commands can be used only in BASIC mode (without a source listing).

The available PL/I subsection commands are:

AUTO	Position at automatic storage.
STAT	Position at static storage.
CTL	Position at controlled storage.

ASM

Subsection commands are available for the Assembler 'program storage and diagnostic' sections. They can be entered when viewing a selected program storage section and also from any Abend-AID report section in combination with the section command PROG(XXXXXXXX).

AMODE31	Displays storage as viewed in 31-bit addressing mode.
A31	Displays storage as viewed in 31-bit addressing mode.
AMODE24	Displays storage as viewed in 24-bit addressing mode.
A24	Displays storage as viewed in 31-bit addressing mode.
AMODE64	Displays storage as viewed in 64-bit addressing mode.
A64	Displays storage as viewed in 64-bit addressing mode.

Option 7—Program Listing

If you select option 7 on the Compuware/VF Output Selection Menu (Figure 8-21 on page 8-22) and only one program exists in the Abend-AID output, the Program Listing section for that program is displayed. The Program Listing section is created for Assembler, COBOL, and PL/I programs processed by the Compuware language processor. By default, it shows source listing statements in the abending program as specified by the global site options SLSLIST (for abending programs) and SLSMLST (for other programs) at abend time. See the *Abend-AID Installation and Customization Guide* for more information. When the program is displayed and the program name is highlighted, this indicates that the field is cursor-selectable and, if selected, will transfer display to the associated program storage ("Option 6 — Program Storage" on page 8-23).

If more than one listing exists in the output, the Compuware/VF Program Listing Selection List (Figure 8-23) is displayed.

Figure 8-23. Compuware/VF Program Listing Selection List

```
COMPUWARE/VF  PROGRAM LISTING SELECTION LIST -------------- ROW 1 TO 2 OF 2
COMMAND INPUT ===> _                                       SCROLL ===> CSR

  PROGRAM / PROCEDURE NAME      LOAD    LANGUAGE     COMPILE DATE  -  TIME

  PAYROLL2                   PAYROLL2  VS COBOL II  23 FEB 1996      13.34.53
  MONTHS2                    PAYROLL2  VS COBOL II  23 FEB 1996      13.34.32
****************************** BOTTOM OF DATA ******************************
```

This screen lists all program listings in the Abend-AID output. The following information is displayed in the selection list:

PROGRAM

The CSECT or PL/I procedure name of the program.

LOAD

The name of the load module containing the program.

LANGUAGE

The language in which the program was written.

COMPILE DATE - TIME

For OS/VS COBOL, VS COBOL II, IBM SAA AD/Cycle COBOL/370, COBOL/390, and PL/I programs, the date and time when the program was compiled.

Selecting a Program Listing

To select a program listing for viewing, enter the line command **S** to the left of the listing name.

To select a specific program listing from the Compuware/VF Output Selection Menu (Figure 8-21 on page 8-22) or within any Abend-AID report section without first displaying the Program Listing Selection List, enter the command 'PLIST**(xxxxxxxx)**' to select listing **xxxxxxxx**.

PLIST Operands

The PLIST command lists the default specified at abend time or the portion specified on the last PLIST command while viewing the current report. To override the default, include one of the following available operands:

PLIST PARA

Lists the COBOL paragraph, PL/I procedure, or 'N' statements preceding the Assembler command containing the current statement.

PLIST LABEL

Lists the labeled portion containing the current statement.

PLIST ALL

Lists the entire COBOL Procedure Division, PL/I, or Assembler listing.

PLIST nn

Lists nn statements before and after the abending statement.

Option 8—Data Management Control Blocks

Selecting option 8 from the Compuware/VF Output Selection menu (Figure 8-21 on page 8-22) shows file and current data management control block information. If only one file exists in the output, that file will be displayed. If more than one file exists in the output, the Compuware/VF File Summary Selection List (Figure 8-24) is displayed.

Figure 8-24. File Summary Selection List

```
COMPUWARE/VF  FILE SUMMARY SELECTION LIST  ------------------- Row 1 to 7 of 7
COMMAND INPUT ===>                                            SCROLL ===> PAGE

  S - view file control blocks  E - File-AID Edit  B - File-AID Browse
                                (except JES spooled datasets)
  DDNAME  DEVICE  UNIT VOLSER DISP AMTYPE EXCPS   ------ DCB INFORMATION ------

  SYSPRINT 3390   021A PRD944 SHR   QSAM  2        (DSORG=PS,RECFM=FB,
                                                   LRECL=80,BLKSIZE=3120)
  FILE2                       SHR   VSAM  2        RECORG=KSDS,MAXRECL=394
  FILE1    3390   021A PRD944 SHR   QSAM  2        (DSORG=PS,RECFM=FB,
                                                   LRECL=80,BLKSIZE=3120)
  SYSOUT   JES2 spooled       MOD   QSAM  4        (DSORG=PS,RECFM=FBA,
                                                   LRECL=121,BLKSIZE=12100)
****************************** Bottom of data ******************************
```

This screen lists all files in the Abend-AID report. The following information is displayed in the selection list:

DDNAME	The ddname from the JCL.
DEVICE	The physical device type.
UNIT	The physical unit address.
VOLSER	The volume serial identification.
DISP	The disposition of the dataset.
AMTYPE	The access method.
EXCPS	The value of the EXCP field in the DCB (block I/O) or spool record count.
DCB INFORMATION	The DSORG, RECFM, LRECL, and BLKSIZE parameters, if available.

Note: The **DEVICE/UNIT/VOLSER** will be replaced with a spool message if the file is spooled.

Selecting Files on this Screen

To select a file for viewing, enter the line command **S** to the left of the file ddname.

To select a specific file from the Output Selection Menu or within any Abend-AID report section, without first displaying the File Summary Selection List when more than one file exists in the output, enter the primary command 'FILE**(***xxxxxxxx***)**' for a specific file ddname *xxxxxxxx*.

Accessing Compuware File-AID

Two File-AID options appear at the top of the Data Management Control Blocks screen: Edit and Browse. If File-AID/MVS is currently installed at your site, entering the **E** (EDIT) or **B** (BROWSE) line commands displays a File-AID Edit or Browse screen containing that report. Figure 8-25 displays a File-AID/MVS Edit screen.

File-AID can also be invoked when specific file information is displayed. In this case, enter the EDIT or BROWSE command at the command prompt to display the File-AID Edit or Browse screen.

Notes:

1. To access File-AID from the Compuware/VF under Option 8, File-AID/MVS releases 8.5 or more current must be installed. If your programs contain DB2 code, or you wish to view the information under Option 11, File-AID for DB2 V3R9 or more current must be installed. File-AID for IMS 4.7 with product update F1 or more current must be installed in order to view the information under Option 9.
2. For use with any of the File-AID versions, Abend-AID 9.1 or more current must also be installed.
3. Various error messages may be displayed if the proper release levels of File-AID are not installed and accessible during view time. If a File-AID error message is received within the Compuware/VF, verify that the proper File-AID libraries are allocated at TSO logon or are allocated within the REXX or CLIST that is being used to invoke the Viewing Facility. Please see Step 7. "Establish Access to File-AID" in the "CSS Customization" chapter of the *Enterprise Common Components Installation and Customization Guide* for additional information.

Figure 8-25. File-AID/MVS Invoked from Compuware/VF

```
File-AID - Edit - ABCD123.SLS.JCL(PRINT3) - 01.99 ----------------------------
COMMAND ===>                                              SCROLL ===> PAGE
FA223- Records read = 21, selected = 21, error records skipped = 0
==MSG>   -NOTE-    CAPS MODE IS "ON"
==MSG>   -NOTE-    STATS MODE IS "ON"
000001 //CWPIRTOD JOB  ('OLPBAS5.ODEV',M15),SMITH,
000002 //       CLASS=P,MSGCLASS=R,NOTIFY=ABCD123
000003 /*JOBPARM ROOM=M15
000004 //*
000005 //*******************************************************************
000006 //*     ---------- PRINT THE MEMBER ----------                      *
000007 //*******************************************************************
000008 //PRINTER  EXEC  PGM=IEBGENER
000009 //*XP      OUTPUT DEST=CWX3835,COPIES=1,
000010 //EXP      OUTPUT DEST=PTRMID,COPIES=1,
000011 //             PAGEDEF=CPS6,CHARS=GT10    PORTRAIT  - 6 LPI 77 WIDE
000012 //*            PAGEDEF=CPS8,CHARS=GT12    PORTRAIT  - 8 LPI 92 WIDE
000013 //*            PAGEDEF=CLS8,CHARS=D226    LANDSCAPE - 8 LPI 143 WIDE
000014 //*            PAGEDEF=CPSC2,CHARS=GT20   TWO UP SIMPLEX
000015 //*
000016 //SYSUT2   DD  SYSOUT=G OUTPUT=*.EXP
000017 //SYSIN    DD  DUMMY
000018 //SYSPRINT DD  SYSOUT=*
Enter FMT for formatted mode, VFMT for vertical format, HEX ON  for Hex
```

Option 9 — IMS Diagnostic Information and PCBs

Abend-AID for IMS information. File-AID for IMS 4.7 with product update F1 or more current must be installed to view information under this option.

Option 10 — CA-IDMS Diagnostic and Status Information

Abend-AID for IDMS information.

Option 11 — DB2 Diagnostic and Status Information

Abend-AID for DB2 information. File-AID for DB2 V3R9 or more current must be installed to view information under this option.

Option 12 — Current COBOL Sort Information

Current sort record for internal COBOL sorts.

Option 13 — Creating COBOL Data Locator Reports

Selecting option 13 (DATALOC) generates a Data Locator Report (Figure 8-26). To create this report, Compuware/VF conducts a search of the Abend-AID report using current user-defined patterns. The Data Locator report lists all variables that adhere to these patterns.

Figure 8-26. Data Locator Report

```
COMPUWARE/VF ---------------------------------------------------------------
COMMAND INPUT ===> _                                        SCROLL ===> CSR
                                                   Formatting has completed
'CX.DUMP.DDIO'           REPT.71   JOBNAME1  J06619
 Program: MLECOB          Compile Date: 07 JAN 2002    YEARWINDOW(1971)

 '@' in column    36 denotes field with Data Locator PICTURE match.
 '!' in column    36 denotes field with Data Locator DATE    match.
 '#' in column    36 denotes field contains data not matching PICTURE clause.

   Language Processor Interface Release      = 07.06.00
   Language Processor Post Processor Release = 07.06.00


             Working-Storage Section


Level/Field Name                Picture/Type Offs        Value
------------------------------  -----------------   ----+----1----+----2
77 CURRENT-YEAR               #  99                  SPACES
77 DAT-DATA-MATCH             !  9(5)                99366
77 DUE-YEAR                      99                  05
              DATE FORMAT    YY       YEAR(2005)
77 END-YEAR                      9999                2005
```

Note: Data Locator is available for COBOL programs **only**.

Julian date fields containing the MLE DATE FORMAT clause are displayed as Julian and Gregorian dates.

How Data Locator Works

The first time that a Data Locator report is generated, default variable patterns are used to generate the report. Each pattern consists of one of the following:

STRING

Selects all variables with COBOL data-names containing **string**.

MLEDATE

Selects all variables containing the COBOL MLE DATE FORMAT **clause**.

DATE

Selects all variables with values matching **date**, even if these dates are not MLE dates.

PICTURE

Selects all variables with a picture clause that matches **picture**.

To view the complete syntax of the variable patterns listed above and edit the patterns, enter the FINDDATA command at the command prompt while a Data Locator report is displayed.

From the sample screen below (see Figure 8-27), you can see that you need to type FINDDATA on the command line. This will allow you to change the selection criteria.

Figure 8-27. Default Variable Pattern for a Data Locator Report

```
COMPUWARE/VF ------------------------------------------------ Invalid Command
COMMAND INPUT ===>                                              SCROLL ===> PAGE
                                                     Formatting has completed
'CX.VSAM.DUMP.DDIO'       REPT.(M) 18  JOBNAME1  J25011
 Pgm: PRICECHK                         Comp. Date: 29 AUG 2000
               Use "FINDDATA" command to change selection criteria
          ***************************************************************
          *         Data analysis for program - PRICECHK                *
          *      Compiled on   29 AUG 2000  at                          *
          *                                                             *
          *                                                             *
          *                                                             *
          *      MLEDATE: ON         YEARWINDOW: N/A                    *
          *                                                             *
          *      Substrings for inclusion in this section:              *
          *                                                             *
          *  BORN                           BIRTH                      *
          *  DATE                           EXPIRY                     *
          *                                                             *
          *                                                             *
          *      Date values to be flagged:                             *
          *                                                             *
          *  98000                          99000                      *
          *  99365                          99366                      *
          *                                                             *
          *                                                             *
          *      PICTURE clauses to be flagged:                         *
          *                                                             *
          *  99/99/9999                     99/99/99                   *
          *                                                             *
          *                                                             *
           ***************************************************************

 ' @' in column   36 denotes field with Data Locator PICTURE match.
 ' !' in column   36 denotes field with Data Locator DATE    match.
 ' #' in column   36 denotes field contains data not matching PICTURE clause.

****************************** Bottom of data ********************************
```

After you type FINDDATA and press enter, the screen below will appear. From this source, you can see that the lines correlate to the screen above. After you change the values below, and press PF3 to Exit, the changes will be reflected in the screen above.

Figure 8-28. FINDDATA Source Screen

```
File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help

 EDIT       SYS01193.T160317.RA000.CX.R0125827              Columns 00001 00064
 Command ===>                                                  Scroll ===> PAGE
 ****** ***************************** Top of Data ******************************
 000001 MLEDATE(ON)
 000002 STRING(BORN)
 000003 STRING(BIRTH)
 000004 STRING(DATE)
 000005 STRING(EXPIRY)
 000006 DATE(98000)
 000007 DATE(99000)
 000008 DATE(99365)
 000009 DATE(99366)
 000010 PICTURE(99/99/9999)
 000011 PICTURE(99/99/99)
 ****** **************************** Bottom of Data ****************************
```

Printing Data Locator Reports

To direct Data Locator Report output to a local printer or dataset for hardcopy purposes, use the LPRINT command from the command prompt.

Note: Data Locator Reports are not part of the standard batch Abend-AID report set. The report can only be produced online, and the LPRINT command is the only method of producing a hardcopy version of the complete report.

Option 14 — Abend-AID for WebSphere MQ

Displays either MQSeries batch or MQSeries IMS information created by Abend-AID for WebSphere MQ.

Option 15 — LE Information

Displays the following LE information:

- LE run-time options at the time of the error indicates how the option was set and from where it was set
- Provides a complete validation and analysis of LE heap storage
- Key LE control blocks are displayed.

Option 16 — Abend-AID Termination Information

Displays Abend-AID termination information.

Mismatched Source Listing Warnings

A mismatched source listing warning will be displayed when the following conditions are true:

- You entered YES in the **Show source warning** field on the Compuware/VF Entry Panel (Figure 8-2 on page 8-3).

  AND

- The portion selected for viewing contains information for the Assembler, COBOL, or PL/I program that does not match the date and time of the source listings with the same program name and language in the specified source listing file.

  OR

- The compiler option, LANGUAGE, which specifies the program language, does not match.

The exact warning depends on the nature of the discrepancy.

Mismatched Date and Time

If a discrepancy is detected that is strictly limited to a difference in the date and time, the warning shown in Mismatched Date and Time Warning (Figure 8-29) is displayed.

Figure 8-29. Mismatched Date and Time Warning for COBOL and PL/1

```
COMPUWARE/VF ----------------------------------------------------------------
COMMAND INPUT ===>

'CX.CSS79.QA.REPTFILE'  REPT.4  JOBNAME1  J31098

     * * * * * * * * * * *    W A R N I N G    * * * * * * * * * * * *
     *       Program                           Most recent listing      *
     *       ------------------------------------------------------- *
     * Name: COBSAMP                           COBSAMP                  *
     * Load: COBSAMP                                                    *
     * Date: 07 DEC 2000                       27 DEC 2001              *
     * Time: 11:22:20                          08:07:38                 *
     *                                                                  *
     * The program date and time stamp does not match a source listing.*
     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 Type an S next to the desired option and press ENTER.

 _ Use the MOST RECENT source listing shown above.
 _ Display available source listings.
 _ Retrieve a source listing by submitting background JCL.
 _ Provide BASIC support without a source listing.

 Enter END command to exit.
```

From this screen, you may choose one of the following:

- Use the most recent source listing shown above.
- Display available source listings.
- Retrieve a source listing by submitting background JCL.
- Provide basic support without a source listing.
- Exit the selected section.

Mismatched Structure for PL/I

The warning shown in Mismatched-Structure Warning (Figure 8-30) is displayed when the following conditions are true:

- The program is a PL/I program.

  AND

- One of the following is also true:
  - One or more procedure names or BEGIN blocks on the calling chain do not exist in the source listing.

    OR

  - A discrepancy is noted in the offset from the beginning of the compile unit.

Figure 8-30. Mismatched-Structure Warning for PL/I

```
COMPUWARE/VF ------------------------------------------------------------------
 COMMAND INPUT ===>

 'CX.CSS79.QA.REPTFILE'  REPT.1  JOBNAME1  J20788

      * * * * * * * * * * * *    W A R N I N G    * * * * * * * * * * * *
      *                                                                   *
      *         An appropriate source listing could not be found.         *
      *         Either:                                                   *
      *           1. a source listing for program PLIA5    does not       *
      *              exist, or                                            *
      *           2. a source listing exists, but it does not match       *
      *              the compiler language and release of the program     *
      *              in the Abend-AID report.                             *
      *                                                                   *
      * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


    Type an S next to the desired option and press ENTER.
        _ Retrieve a source listing by submitting background JCL.
        _ Provide BASIC support without a source listing.


   Enter END command to exit.
```

From this screen, you may choose one of the following:

- Retrieve a source listing by submitting background JCL.
- Provide Basic support without a source listing.
- Exit the selected section.

Mismatched Structure for Assembler

Since Assembler programs often contain more than one CSECT (named program section), it may be that the name of the CSECT in which an abend occurred bears no relation to the name of the source listing in DDIO. In this case, the Compuware Viewing Facility presents you with a list of all Assembler programs in any of the source listing DDIO files you are using (see Figure 8-31).

Figure 8-31. Assembler Mismatch List

```
COMPUWARE/VF ------------------------------------------------  ROW 1 TO 3 OF 3
 COMMAND INPUT ===>                                            SCROLL ===> CSR

     CSECT Name: EUR001
  Wildcard Mask: ********
      Assembled: 16 AUG 2000            Load module: EUR001

    The CSECT name and date do not uniquely match a source listing.

  _ Retrieve a source listing by submitting background JCL.
  _ Provide BASIC support without a source listing.

    Type an S next to the correct source listing below and press the ENTER key.
                 ASSEMBLER SOURCE LISTING DATASET DIRECTORY
    PROGRAM    LIST NUMBER     ASMB DATE     TIME    RC   LANGUAGE    SIZE(K)
   ---------- 'EFHRIP0.LISTFILE' -------------------------------------------
   CCADEMO   (L)        19    08 MAR 2002  10.19     00   HLASM R2    2196
   PTFBLD               15    18 FEB 2002  09.02     00   HLASM R2      96
   ASMCRGD              14    28 JAN 2002  13.51     00   HLASM R2     147
   AA01DEMO  (M)        13    23 JAN 2002  16.44     00   HLASM R2     230
   LITXNY                9    02 JAN 2002  12.17     00   HLASM R2      75
 ****************************** BOTTOM OF DATA ********************************
```

From this screen, you may select the source listing that contains the “CSECT Name” displayed on the top left of the screen.

On-the-Fly Language Processing Support

When viewing an Abend-AID report, sometimes no source listing member is available in any of the selected Source DDIO files or Source Database files (No Source). Then report viewing will not have source support with all of its advantages. Additionally, sometimes a source listing member is available, but it does not “match” the same parameters as the Abend-AID report (Mismatch). That is, the time of the compilation does not match the abending load module, or perhaps the source listing member was created with a different version of COBOL compiler (i.e., not the same as used for the abending load module).

When these situations are detected, on-the-fly language processing support allows you to optionally invoke language processing immediately. After the processing is completed, the Abend-AID report can be reselected and full source support is then available.

Whenever a No Source or Mismatch situation occurs, one of the following two panels appears in the viewing facility (Figure 8-32 or Figure 8-33). Each of these panels existed previously (before CSS 7.9), but an extra option now appears on the panel (shown in bold) for the on-the-fly options.

Panel AAAEPNSL (shown in Figure 8-32) and AAAEPMIS (shown in Figure 8-33) both provide a Retrieve option.

Figure 8-32. Source Not Found Warning Panel

```
COMPUWARE/VF ----------------------------------------------------------------
 COMMAND INPUT ===>

 'CX.RPTFILE'  REPT.92  EFHRIPOC  J20788

      * * * * * * * * * * *    W A R N I N G    * * * * * * * * * * *
      *                                                               *
      *         An appropriate source listing could not be found.     *
      *         Either:                                               *
      *           1. a source listing for program COMP5    does not   *
      *              exist, or                                        *
      *           2. a source listing exists, but it does not match   *
      *              the compiler language and release of the program *
      *              in the Abend-AID report.                         *
      *                                                               *
      * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


    Type an S next to the desired option and press ENTER.
        _ Retrieve a source listing by submitting background JCL
        _ Provide BASIC support without a source listing


   Enter END command to exit.
```

Figure 8-33. Program Date and Time Do Not Match Source Listing

```
Panel AAAEPMIS.
COMPUWARE/VF ------------------------------------------------------------------
COMMAND INPUT ===> _

'EFHRIPO.RPTFILE'  REPT.14  EFHRIPOC  14078


     * * * * * * * * * * *    W A R N I N G    * * * * * * * * * * * *
     *       Program                          Most recent listing      *
     *       ------------------------------------------------------- *
     * Name: COBMIS                          COBMIS                   *
     * Load: COBMIS                                                   *
     * Date: 01 MAY 2002                     30 APR 2002              *
     * Time: 13:52:07                        09:37:44                 *
     *                                                                *
     * The program date and time stamp does not match a source listing.*
     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 Type an S next to the desired option and press ENTER.

_  Use the MOST RECENT source listing shown above.
_  Display available source listings.
_  Retrieve a source listing by submitting background JCL.
_  Provide BASIC support without a source listing.
```

Panel AAAEMALS (shown in Figure 8-34) is a special panel used by Assembler programs that have a “mismatch.”

Figure 8-34. Assembler Mismatch Panel

```
COMPUWARE/VF ----------------------------------------------  ROW 1 TO 28 OF 31
 COMMAND INPUT ===>                                            SCROLL ===> CSR

     CSECT Name: EUR001
  Wildcard Mask: ********
      Assembled: 16 AUG 2000            Load module: EUR001

    The CSECT name and date do not uniquely match a source listing.

  _ Retrieve a source listing by submitting background JCL.
  _ Provide BASIC support without a source listing.

    Type an S next to the correct source listing below and press the ENTER key.
                 ASSEMBLER SOURCE LISTING DATASET DIRECTORY
    PROGRAM    LIST NUMBER     ASMB DATE     TIME    RC   LANGUAGE    SIZE(K)
   ---------- 'EFHAXSO.TEST.DDIO' ----------------------------------------
   X80102B   (L)      3537    09 NOV 2001  12.20     00   HLASM R4     399
   TEMPNAME  (L)      3344    10 SEP 2001  13.33     00   HLASM R4     160
   TEST1     (L)      3179    22 JUN 2001  11.35     00   HLASM R4     160
   VIBAIITF  (L)      3046    28 MAR 2001  13.07     00   HLASM R4     559
   CWDEMASM  (L)      2612    14 JUN 2000  20.08     04   HLASM R3     240
   ASM03T    (L)      2651    08 JUN 2000  15.47     00   HLASM R3      80
   ASMPRG1   (L)      2330    13 MAR 2000  12.14     00   HLASM R3      80
   AIBHLP00  (L)      1023    T PAGO N     10.51     00   HLASM R2     719
   BXPLDBRT  (L)      1884    03 AUG 1999  13.39     04   ASMH         399
   L87C006   (L)      1598    08 JUL 1999  16.37     00   HLASM R2     399
```

You can select “retrieve” to begin on-the-fly language processing, or select BASIC support, on one of the panels above.

If BASIC support is selected, no source support will be available, and the abend report will be viewed without any source support. The next panel to be displayed will be the normal Abend-AID Analysis of Errors section without source support.

If the “Retrieve” option is selected on either of these panels, you will be presented with additional panels that conduct the on-the-fly language processing sequence. The next panel is:

Figure 8-35. Source Listing Retrieval Menu

```
COMPUWARE/VF ------------------------------------------------------------------
 COMMAND INPUT ===>


                          SOURCE LISTING RETRIEVAL MENU

    Type an S next to the desired option and press ENTER to generate JCL:


        _ A compiler listing exists and needs to be converted to a
          usable Compuware Source Listing format.


        _ No compiler listing exists.  One needs to be created by
          compiling into a Compuware Source Listing format.


                            Enter END command to exit.
```

You have two selection options on this panel. Each option is essentially a choice between language postprocessing only, or compile and process.

The first option is used when a compiler listing exists, but it is not stored in a Compuware DDIO file. The listing may exist in a sequential or partitioned dataset, or possibly in a listing database. you only need to postprocess this listing with the Compuware Language Processor in order to create a source listing member in a DDIO file.

The second option is used when there is no compiler listing available. When this occurs, a listing must be re-created by recompiling the source program. Choosing this option will select the CSS language preprocessor, which will compile the source program and create a resulting source listing member into a DDIO file.

Each of the two options above provides two additional choices. The additional choices focus around using a customer-supplied procedure to invoke language postprocessing, or language preprocessing (compile and process). The panel with the additional choices is shown in Figure 8-36:

Figure 8-36. Select Postprocess Procedure Screen

```
COMPUWARE/VF ------------------------------------------------------------------
COMMAND INPUT ===>


                            POSTPROCESS PROCEDURE

    Type an S next to the desired option and press ENTER to generate JCL:


        _ Execute a customer defined procedure to postprocess.


        _ Execute CSS Utilities to begin postprocessing.


                            Enter END command to exit.
```

Whenever the first option is chosen, a TSO CLIST is invoked. Some customers have already prepared customer procedures that they use to perform language processing tasks. If they choose, they can have the viewing facility call their procedure directly by selecting the first option. In order for this option to work properly, your CLIST must be available in the proper library and set up to be called. Compuware will also provide working sample CLISTs that illustrate how a customer CLIST could be created. The

sample CLISTs are called CXOTFPRE (for preprocessing-oriented procedures) and CXOTFPST (for postprocessing-oriented procedures).

If the second option is chosen, then no customer-supplied CLIST exists or is desired, so the standard CSS Utilities will be invoked to perform preprocessing or postprocessing. If you choose this option, you will be taken directly to one of the existing CSS Utilities panels to begin tailoring your language processing options. For clarity, an example of one of the CSS Utilities panels is shown in Figure 8-37:

Figure 8-37. CSS Utilities - Load Existing Listing Screen

```
CXUEPOP1 ---- COMPUWARE CSS UTILITIES -- LOAD EXISTING LISTING --------- V8.0.0
 COMMAND ===>

 Primary Commands: blank (Process Listing)   Browse  Edit  SEtup


 Compiler Listing  ===>
      Preparation  ===> BATCH         Batch/Editjcl
         Language  ===> COBOL         COBOL/PLI/ASSEMBLER



 DDIO Listing File  ===> LISTFILE
  Options Dataset  ===>
   Enhanced FIND  ===> N      (Y/N)
```

After you complete the customer-supplied language processing actions, or use the CSS Utilities panels, you will be returned to the following panel (see Figure 8-38):

Figure 8-38. Post-Submission Selection Menu

```
COMPUWARE/VF ------------------------------------------------------------------
COMMAND INPUT ===>


                          Post-Submission Selection MENU

   If you submitted JCL, it will execute in the background.  You will be
   informed of job completion because a JCL JOBCARD NOTIFY= was specified.


   Until this JCL completes, you have the following options:

   Type an S next to the desired option and press ENTER:

       _ Provide BASIC support without a source listing.
       _ Return to the OUTPUT SELECTION MENU to select a different
         section of the same report.
       _ Return to the Abend-AID DATASET DIRECTORY to select a different
         report.


   Enter END command to exit.
```

You have three choices on the post-submission selection menu. Presumably JCL has been submitted to the system for execution that will create a source listing member. If you submitted JCL, then there will be a delay while the JCL executes. In the meantime, you can select any of the options available, or wait for JCL completion. If you did not submit any JCL, the three choices are nevertheless still valid.

The first choice is to continue viewing Abend-AID with BASIC support (that is, no source support). Later, you can reselect the abend report on the report selection menu to gain access to source after the JCL execution is completed.

The second choice allows you to select a different section of the same Abend-AID report while waiting for JCL completion.

The third choice allows you to go immediately to the report selection menu and select a different Abend-AID report while waiting for JCL completion.

Displaying Available Source Listings

If you choose to display available source listings, the Source Listing Dataset Directory Screen (Figure 8-39) is displayed.

Figure 8-39. Source Listing Dataset Directory Screen

```
COMPUWARE/VF ---------------------------------------------  ROW  1 TO 3 OF 3
COMMAND INPUT ===>                                            SCROLL ===> PAGE
                    SOURCE LISTING DATASET DIRECTORY
   PROGRAM    LIST NUMBER   COMP DATE    TIME    RC   LANGUAGE    SIZE
  ---------- 'CWX0236.TECH.FIL1169.SLS2' -------------------------------
  WTP#1074  (L)        16   05 MAY 1996  16:16:18 00   PL/I V2    48 KB
  WTP#1074              8   29 MAR 1996  09:16:26 00   PL/I V2    32 KB
  ---------- 'CWX0236.TECH.FIL1169.SLS1' -------------------------------
  WTP#1074  (L)        17   04 MAY 1996  10:49:45 00   PL/I V2    82 KB
  WTP#1074             16   03 MAR 1996  09:16:26 00   PL/I V2    82 KB
  ---------- 'CWX0236.BD60.SLSP' ---------------------------------------
  WTP#1074  (L)        19   05 MAY 1996  16:16:18 00   PL/I V2    44 KB
  WTP#1074  (L)        12   04 FEB 1996  10:49:45 00   PL/I V2    40 KB
  WTP#1074  (L)        10   02 FEB 1996  09:16:26 00   PL/I V2    32 KB
****************************** BOTTOM OF DATA ******************************
```

The source listings are arranged by source listing file. A separator line that includes the name of the source listing file precedes the source listings for that source listing file. The Source Listing Dataset Directory Screen (Figure 8-39) displays the following information:

PROGRAM:

Lists the members (source listings) for each source listing file.

LIST NUMBER:

The number assigned by the language processor to identify the source listing within each source listing file. The source listings within each source listing file are numbered sequentially beginning at 1 and continuing until the source listing file is reformatted. The following status indicators also appear, if applicable, in the **List Number** column.

- ***I** Source listing is incomplete.
- ***IL** Source listing is incomplete and locked.
- ***IM** Source listing is incomplete and manually locked.
- **(L)** Source listing is automatically locked.
- **(M)** Source listing is manually locked.

COMP DATE:

The date the program was compiled.

TIME:

The time the program was compiled.

RC

The Compuware language processor return code.

LANGUAGE

The program language compiler and version. The following codes can be found in the LANGUAGE column:

ASMF	Assembler F
ASMH	Assembler H
HLASM	High Level Assembler 1.1
HLASM R2	High Level Assembler 1.2
HLASM R3	High Level Assembler 1.3
HLASM R4	High Level Assembler 1.4
OSVSCOB	OS/VS COBOL
OSVSCOBOPT	OS/VS COBOL/CA-Optimizer
C/ESA V3R2	"C" for ESA V3R2
C/390 V1R2	OS/390 "C" V1R2
C/390 V1R3	OS/390 "C" V1R3
C/390 V2R4	OS/390 "C" V2R4
C/390 V2R5	OS/390 "C" V2R5
C/390 V2R6	OS/390 "C" V2R6
C/390 V2R9	OS/390 "C" V2R9
C/390 V2RA	OS/390 "C" V2R10
COBIIR2	VS COBOL II Release 2
COBIIR2OPT	VS COBOL II Release 2/CA-Optimizer II
COBIIR3	VS COBOL II Release 3
COBIIR3OPT	VS COBOL II Release 3/CA-Optimizer II
COBIIR4	VS COBOL II Release 4
COBIIR4 JA	VS COBOL II Release 4 with Japanese Language Support
COBIIR4OPT	VS COBOL II Release 4/CA-Optimizer II
COBII JA	VS COBOL II with Japanese Language Support
COBOL/370	IBM SAA AD/Cycle COBOL/370
COBOL/370 JA	IBM SAA AD/Cycle COBOL/370 with Japanese Language Support
COBOL/370OPT	IBM SAA AD/Cycle COBOL/370/CA-Optimizer II
COBOL/390	COBOL for OS/390 and VM
COBOL/390M	COBOL for OS/390 and VM with MLE (Millennium Language Extensions)
COBOL/MVSM	COBOL for MVS and VM with MLE
COB/370JA	COBOL/370 Japanese Language Support
COB/390OPM	COBOL/390 MLE CA-Optimizer
COB/390OPT	COBOL/390 CA-Optimizer
COB/MVS	COBOL for MVS and VM
COB/MVS JA	COBOL for MVS and VM with Japanese Language Support
COB/MVSOPM	COBOL/MVS MLE CA-Optimizer
COB/MVSOPT	COBOL/MVS CA-Optimizer
COBOL Z/OS	Enterprise COBOL for z/OS and OS/390.
COBOL Z/OSJA	Enterprise COBOL for z/OS and OS/390 with Japanese language support.
COB/Z V3R2	Enterprise COBOL for z/OS and OS/390 Release 3.2

COB Z/OSJA	Enterprise COBOL for z/OS and OS/390 Release 3.2 with Japanese Language Support
PL/I V1R5	PL/I V1
PL/I V2	PL/I V2
PL/I AD V1	PL/I for MVS and VM
PL1VA V2R2	PL/I VisualAge V2R2
PL1EN V3R1	Enterprise PL/I for z/OS
PL1EN V3R2	Enterprise PL/I for z/OS

SIZE:

The number of bytes (KB) allocated for the source listing in the source listing file.

Line Commands from Source Listing Dataset Directory Screen

From the Source Listing Dataset Directory Screen (Figure 8-39 on page 8-38), you can enter the following line commands:

B Browses the source listing.

D Deletes the source listing from the source listing file. If you entered YES in the Confirm delete field on the Compuware/VF Entry Panel (Figure 8-2 on page 8-3), a Confirm Delete screen will be displayed before allowing you to delete the source listing. If the member is locked (automatically or manually), the member is unlocked and the source listing deleted.

L Locks the source listing so it cannot be automatically deleted when the source listing file formatted with AUTODELETE=YES, AUTODELETE=DUPS, or AUTODELETE=STAGED becomes full. Manual locks appear with a status indicator of (M) on the Source Listing Directory screen (Figure 8-39 on page 8-38).

P Prints the source listing using the hardcopy facility. If you entered YES in the Show print setup field on the Compuware/VF Entry Panel (Figure 8-2 on page 8-3), the Hardcopy Options Screen (Figure 8-20 on page 8-20) will be displayed.

S Selects the source listing for viewing the selected portion of the Abend-AID report.

U Unlocks the source listing so it can be automatically deleted when the source listing file formatted with AUTODELETE=YES, AUTODELETE=DUPS, or AUTODELETE=STAGED becomes full.

Notes:

1. The commands **D**, **L**, **P**, and **U** are the same as for the standard Dataset Directory Screen.
2. For additional information on the AUTODELETE parameter, refer to:
   - "Command Parameters for DDIO Files" on page 4-6 (for the CWDDSUTL utility),
   - "Command Parameters for Shared Directories and Transaction Databases" on page 5-5 (for the CWFXSDUT utility),
   - "Command Parameters for Shared Directories and Report Files" on page 6-5 (for the CWAASDUT utility),
   - "Command Parameters for Shared Directories and Source Listing Databases" on page 7-6 (for the CWDDLPUT utility).

When multiple line commands are entered on the Dataset Directory Screen, shown as a result of selecting "Display available source listings" from the mismatched source listing warning, they will be executed from top to bottom until the first **S** command is executed. Any remaining line commands will be ignored.

Note: If you select either basic support or a source listing for viewing the selected portion of the Abend-AID report, the selection will remain in effect until you exit the Abend-AID report or enter the SOURCE command.

Source-Not-Found Warning

The warning shown in Source-Not-Found Warning (Figure 8-40) will be displayed when the following conditions are true:

- You entered YES in the **Show source warning** field on the Compuware/VF Entry Panel (Figure 8-2 on page 8-3).

  AND

- The portion selected for viewing contains information for the COBOL or PL/I program that does not have a source listing with the same program name and language as in the specified source listing file.

Figure 8-40. Source-Not-Found Warning

```
COMPUWARE/VF  ----------------------------------------------------------
COMMAND INPUT ===> _

'CWX0236.DD57.RAA1'  REPT.(L) 7  B51115C2 JOB17222

     * * * * * * * * * * * *   W A R N I N G   * * * * * * * * * * * * *
     *                                                                 *
     *         An appropriate source listing could not be found.       *
     *                                                                 *
     *         Either:                                                 *
     *           1. a source listing for program JBERNAT  does not     *
     *              exist, or                                          *
     *           2. a source listing exists, but it does not match     *
     *              the compiler language and release of the program   *
     *              in the Abend-AID report.                           *
     *                                                                 *
     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

  Type an S next to the desired option and press ENTER.
      _ Retrieve a source listing by submitting background JCL.
      _ Provide BASIC support without a source listing.


Enter END command to exit.
```

From this screen, you may select one of the following:

- Submit a background job to retrieve a source listing.
- Continue without source support
- Exit the selected section.

Note: If you select basic support, the selection will remain in effect until you exit the Abend-AID report or enter the SOURCE command.

SOURCE Command

The SOURCE command lets you suspend or reinstate the use of source support.

The following diagram illustrates the syntax for the SOURCE command.

```
                ┌ON──────┐
►►─SOURCE───────┼────────┼──►◄
                ├OFF─────┤
                └MISMatch─┘
```

ON:

Use source support.

OFF:

Temporarily suspend source support.

MISMATCH:

Specifies to the Viewing Facility to display the appropriate mismatched screen(s) the next time the source is merged with the report. This lets you specify the correct source for the Abend-AID listing or Abend-AID report. This parameter is available on the following screens:

- Compuware/VF Output Selection Menu (Figure 8-21 on page 8-22)
- Compuware/VF Program Storage Selection List (Figure 8-22 on page 8-23)
- Compuware/VF Program Listing Selection List (Figure 8-23 on page 8-25)
- any screen displaying an Abend-AID report section.

When viewing a panel from the Abend-AID report, the mismatch screen(s) display immediately. When viewing the Compuware/VF Output Selection Menu or the selection lists, the mismatch screen(s) display after you merge the source with the report.

You can enter the SOURCE MISMATCH command from the Compuware/VF Output Selection Menu or from any Abend-AID report section. Once entered, this command remains in effect only while viewing the current report.

If the SOURCE MISMATCH command is entered while viewing a section, the section will be redisplayed in the requested format and repositioned at the last entry point. For example, if the last entry was DIAGS, then the display will be positioned at DIAGS in the requested format regardless of the section displayed at the time the SOURCE MISMATCH command was entered.

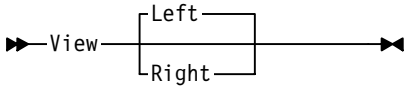
The SOURCE MISMATCH command cancels selections made on the warning screens.

VIEW (V) Command

The VIEW command controls the display of Assembler source lines in an Abend-AID report that is merging Assembler source with Abend-AID text. The View setting is retained throughout the session; it is not saved across sessions. Each new session begins with View Left in effect.

Note: The View setting affects only the display of source lines in Assembler.

The following diagram illustrates the syntax for the VIEW command.



Left:

(Default) This command may be entered as **V L**. When View Left is in effect, the Assembler source line display includes the Assembler line number, the offset of the instruction, the object code of the instruction, any label on the instruction, and as much of the source code as fits on the screen. For example:

```
33  000018 1821       +SETPLIST LR  R2, R1           Caller's plist
```

Right:

This command may be entered as **V R**. When View Right is in effect, the display includes the Assembler line number, any label on the instruction, and the source code out to and including column 72. For example:

```
33+SETPLIST LR  R2, R1         Caller's plist and interface block to R2
```

Browsing Abend-AID Reports

After you select a section of the Abend-AID report, the beginning of that section is displayed. You can then scroll through any viewable portion of the Abend-AID report using the FIND and FS commands or standard ISPF scroll commands. The viewable portion of the Abend-AID report depends on the type of report file:

Abend-AID

If you select any section number from 0 through 5, all existing sections with numbers 0 through 5 can be viewed. If you select any section number greater than 5, only the selected section can be viewed.

To view a section of the Abend-AID report that is not currently formatted for viewing, enter the section number or abbreviation in the **COMMAND INPUT** field, or return to the Compuware/VF Output Selection Menu (Figure 8-21 on page 8-22) by entering the END command.

FIND and FS Commands

After an Abend-AID report section is selected, the FIND and FS commands are valid in the **COMMAND INPUT** field of the Abend-AID report output screen.

The FIND command is similar to the ISPF FIND command. Compuware/VF detects the current cursor position and conducts a search for the specified string starting from this position.

- If an occurrence of the string is found on the current screen, the cursor is moved to that occurrence.
- If an occurrence of the string is not found on the current screen but is found on a subsequent screen, the line containing the string is repositioned to appear near the top of the screen.
- If the string is not found, the search stops at the end of the viewable portion of the report and the message *BOTTOM OF DATA REACHED* is displayed.

The FS command is similar to the FIND command, but is case-sensitive. Only strings with matching uppercase and lowercase characteristics will be found. For example, specifying "REPORT" (all uppercase characters) will not find occurrences of "Report" (mixed case) or "report" (lowercase). To find these alternative forms of "REPORT", use the FIND command.

Searches may be conducted using any of the following:

F *xxx*	Finds a character string *xxx* within the viewable portion of the report.
F *'xxx yyy'*	Finds a character string *xxx yyy* that contains embedded blanks; the single quotation marks define the limits of the string.
FS *xxx*	Conducts a case-sensitive search for character string *xxx* within the viewable portion of the report.
PF5	Finds the next occurrence of the specified string after a FIND is performed.
FS	Repeats the FS command for the string that was last specified.

Note: Use the FS command if the string contains double-byte characters.

When Compuware/VF is formatting program storage areas (when Program Storage has been selected from the Compuware/VF Output Selection Menu), the status area displays a message that states "formatting still in progress." If you use the FIND or FS command at this point, Compuware/VF searches the currently-formatted data areas for the specified string.

Accessing File-AID from Compuware/VF

CSS enables you to invoke File-AID/MVS, File-AID for DB2, or File-AID for IMS to browse or edit files directly from Compuware/VF. To access File-AID in this manner, complete the following steps:

1. Ensure that one of the following versions of File-AID is installed:
   - File-AID/MVS 8.5 or more current
   - File-AID for DB2 V3R9 or more current
   - File-AID for IMS 4.7 with product update F1.

   and Abend-AID must be installed:
   - Abend-AID 9.1 or more current
2. Ensure that the File-AID libraries are allocated at TSO logon or are allocated within the REXX or CLIST that is being used to invoke the Compuware/VF.
3. Choose one of the following options on the Compuware/VF Output Selection Menu:

Table 8-5. Accessing File-AID

To access	Choose this option
File-AID/MVS	Option 8, Data management control blocks
File-AID for DB2	Option 11, DB2 diagnostic and status information
File-AID for IMS	Option 9, IMS diagnostic information and PCBs

4. Two File-AID options appear at the top of the screen: EDIT and BROWSE. Enter one of the following:
   - If the Abend-AID report contains multiple files and a File Summary Selection list appears (see Figure 8-24 on page 8-26), use the **E** (EDIT) or **B** (BROWSE) line command to display a File-AID Edit or Browse screen containing that report.
   - If specific file information appears, enter the EDIT or BROWSE command at the command prompt to display a File-AID Edit or Browse screen.

Accessing CSS Utilities from Compuware/VF

CSS Utilities walks you step-by-step through CSS Language Processor functions and DDIO file manipulation with easy-to-use panels. Supported Language Processor functions include generating preprocessing and postprocessing JCL. DDIO file functions include file creation and formatting, file deletion, and member manipulation (for example, locking/unlocking members, copying members, and importing/exporting members).

CSS Utilities may be used with the following products:

- Abend-AID
- Abend-AID for CICS (for source listing and DSECT files only)
- XPEDITER/CICS
- XPEDITER/TSO

To access CSS Utilities from Compuware/VF, enter the CSSUTIL command at the command prompt on any Compuware/VF screen. The CSS Utilities - Primary Menu (Figure 3-1 on page 3-2) is displayed.

For a complete description of CSS Utilities functionality, see **Chapter 3, "CSS Utilities"**. To return to Compuware/VF from CSS Utilities, press PF3.

Chapter 9.
COBOL Language Processor

A language processor writes a source listing to a VSAM or sequential DDIO file. For COBOL, the source listing includes a compiler listing and other information about a program. This chapter describes the outputs and commands associated with the COBOL language processor.

Operation

The COBOL language processor reads the compiler listing produced by the COBOL compiler and writes either the compiler listing or an enhanced listing to the DDIO file. The source listing can then be used to provide source support during a debugging or diagnostic session using any of the following products:

- Abend-AID
- Abend-AID for CICS
- XPEDITER/CICS
- XPEDITER/TSO.

The COBOL language processor can be run either as a preprocessor or as a postprocessor. Compuware recommends that you use the preprocessor instead of the postprocessor. The postprocessor should only be used in situations where compiled source listings have been stored in machine-readable format.

Preprocessor

Some information about a program module is not always available from the compiler listing. The preprocessor, in addition to gathering information from the compiler listing, gathers information from SYSIN and SYSLIB. The preprocessor provides additional functionality for XPEDITER/CICS and XPEDITER/TSO.

Preprocessor Steps

1. Determines the proper compiler options required to process the listing file.
2. Automatically invokes the compiler or assembler to compile or assemble your source program.
3. Writes the listing to the DDIO file. An enhanced listing for COBOL programs can be produced, if desired.

Preprocessor Benefits

- **Better handling of compiler errors**. By using the preprocessor, you can avoid the potential problems of processing a listing with compiler errors. The preprocessor, in conjunction with the CONDDDIO parameter, internally checks the return code from the compiler and doesn't write a DDIO listing that contains errors. For example, if you set the CONDDDIO parameter to 8, the preprocessor will write the compiler listing to the DDIO file unless the compiler return code exceeds 8. The preprocessor can process compiler errors more effectively than the postprocessor can.

- **Capturing of suppressed source code**. When the following parameter is used, sections of source code can be suppressed from the compiler listing:

  For COBOL Copy SUPPRESS

  The preprocessor captures this information from SYSIN and SYSLIB.

- **Automated compiler options**. The postprocessor requires that certain compiler options be specified in order to process all needed sections of the compiler listing. The preprocessor can automatically pass the required options to the compiler.

  For a list of the required compiler options, refer to "Required Options for Supported COBOL Compilers" on page 9-9.

- **Simplified JCL**. While the postprocessor requires that you add a step after the compile step, the preprocessor requires only that minor modifications be made to your existing compile step.

Dynamically Allocated Files

The following files are dynamically allocated by the Compuware preprocessor. The attributes can be overridden by specifying a DD in the JCL.

Table 9-1. Files Dynamically Allocated by the Preprocessor

DDname	Attributes
CWPWRKn (0-6)	BLKSIZE=19000 SPACE(TRK,(100,80)) UNIT=SYSDA
SORTWK01	BLKSIZE=19000 SPACE(TRK,(100,80)) UNIT=SYSDA
TEMPLIN	BLKSIZE=3200 SPACE(TRK,(200,100)) UNIT=SYSDA
CWPERRM	SYSOUT=*
SYSPRINT	SYSOUT=*
SYSOUT	SYSOUT=*
CWPPRTI	BLKSIZE=16093 SPACE(TRK,(100,200)) UNIT=SYSDA DISP=MOD
CWPPDS2	SPACE (TRK,(100,80,80)) UNIT=SYSDA

CWPPRTO is dynamically allocated as a work file (for the preprocessor). SYSPRINT is where the listing goes.

Postprocessor

The postprocessor is executed as a single step — can be a separate job. It reads in the listing created from the compiler. Information is gathered from the source listing, XREF, data maps, and object code sections of the listing. This information is stored in a DDIO file member and is used by various Compuware products.

While the preprocessor is the preferred method of loading the compiled listing to the DDIO, there are situations that necessitate use of the postprocessor as a viable alternative. This is commonly found in production environments where the listings may be archived. In order for the listings to populate the DDIO, they must be compiled using the required options specified in "Required Options for Supported COBOL Compilers" on page 9-9. Note that these are usually the default options at many shops.

If the programs contain suppressed source code, certain additional control statements must be added to the language processor.

If the programs are compiled with the COBOL compiler option OFFSET as well as the COBOL compiler option OPT(imize), certain additional control statements must be added to the language processor.

Benefits of Postprocessing

- Exact match of compiled output listings with the executable load module. This is especially important in production environments where time is critical.
- Eliminates the risk of copybook or source code changes prior to recompiling the code since the listing reflects the code that is being executed.
- Significantly less time is needed to process source listings as opposed to recompiling the source code.

Summary of COBOL Language Processor Commands

The following is a list and brief description of the commands available for the COBOL language processor. Each command is discussed in detail later in this chapter.

CICSTEST	Instructs the COBOL language processor to print either a warning or severe message regarding the status of the COBOL compile options for running a program in CICS.
COBOL	Controls the compiler listing destination.
CONDDDIO	***(Can only be used with the preprocessor)*** Specifies that the source listing will be written to the source listing file unless the compiler return code exceeds the maximum value you set.
CONDERRL	***(Can only be used with the preprocessor)*** Indicates that the hard copy listing will contain only error messages and lines in error if the compiler condition code exceeds the maximum.
CONDSTEP	***(Can only be used with the preprocessor)*** Controls the condition code set by the postprocessor when it is invoked by the preprocessor.
DDIO	Identifies the sections of the COBOL compiler listing that are written to the source listing file.
File-overrides	***(Can only be used with the postprocessor)*** Used to override the default file definitions.
INCLUDE	***(Can only be used with the preprocessor)*** Used to include a library or sequential file member that contains another set of language processor options to be processed.
LANGPARM	***(Can only be used with the preprocessor)*** Specifies parameters to be passed to the compiler invoked by the language processor.
LANGUAGE	Specifies the type of compiler for the COBOL language preprocessor and postprocessor.
PRINT	Identifies the sections of the listing that will be printed.
PROCESSOR	Specifies the output parameters of the COBOL language processor.
PROGRAM-NAME	Specifies an override name for members in the source listing file.
SYSTEM	Performs miscellaneous functions.

The commands can be entered in two ways:

- Parameters on the PARM statement of the EXEC — (Overrides input from the CWPPRMO file.)

- Input from the CWPPRMO file — (Comments are not valid and all command strings must be within columns 1 through 71.)

Most commands have parameters and some parameters have values. Some commands don't have parameters but require only a value such as the PROGRAM-NAME command. When including parameters and/or values with a command, the parameters and values must be written immediately after the commands, enclosed in parentheses, and separated by commas.

If conflicting command options are entered, the last option specified is used.

Usage Considerations

This section includes information that you need to know in order to be able to use the COBOL language processor most efficiently.

Language Processor DD Statements

The following table lists the DD statements for the COBOL language processor.

Table 9-2. Language Processor DD Statements.

DDname	Purpose
CWPWRK (0-6)	LP work files
CWPPRMO	LP parameter input dataset
CWPDDIO	Target source listing DDIO file
CWPPRTI	Compiler listing input to postprocessor
CWPPRTO	Compiler listing output from postprocessor
CWPLOAD	Option LOAD or OBJECT module input to postprocessor
CWPDECK	Option DECK module input to postprocessor
CWPERRM	LP Error Messages

Note: CWPLOAD and CWPDECK correspond to the SYSLIN and SYSPUNCH compiler outputs. If you specify compiler option OBJECT, you should use CWPLOAD in your postprocessor JCL. If you specify compiler option DECK, you should use CWPDECK in your postprocessor JCL. Do not feed the SYSLIN output into CWPDECK or the SYSPUNCH output into CWPLOAD. The updated OBJECT contained in CWPLOAD and CWPDECK should be used in the program's linkedit step. The preprocessor does not use CWPLOAD and CWPDECK.

COBOL Samples and Program Names

The following is a list of members in the CSS installation library that contain sample JCL for executing the COBOL language processor:

Table 9-3. COBOL Sample JCL Members

CXCOBPRE	JCL for running the COBOL language preprocessor
CXCOB1	JCL to be added after the compile to process the compiler listing through the COBOL language postprocessor
CXCOB2	JCL to postprocess COBOL compiler listings stored from a previous compile job
CXLPCOBB	Sample COBOL language processor options (batch programs)

Table 9-3. COBOL Sample JCL Members

CXLPCOBC	Sample COBOL language processor options (CICS programs)

The following program names are used for the COBOL language processor:

Table 9-4. COBOL Program Names

CWPCMAIN	COBOL preprocessor program
CWPCDRVR	COBOL postprocessor program

Other Usage Considerations

Using Alternative Reserved Word Tables

The alternative reserved word feature of the COBOL language processor lets you create synonyms for compiler reserved words, package them into a load module table for later use, and instruct the compiler to allow COBOL programs to use these synonyms. For example, you could set CALCULATE to be a synonym for COMPUTE. The compiler will then recognize CALCULATE and treat it in every instance as if it were a COMPUTE word.

The WORD option is not supported when $NO or $NG is specified. Since $NO bypasses the dummy compile and flexible options processing, CSS is unable to detect the presence of the WORD option.

To invoke the WORD option for the CSS COBOL preprocessor, use one of the following methods:

The WORD command can only be used in the CSS preprocessor.

- Add WORD(*tabl*) to the list of parameters (PARM=) included on the EXEC statement that executes CWPCMAIN.
- Include LANGP(WORD(*tabl*)) on a separate line in the dataset specified by the CWPPRMO DD statement.

The *tabl* variable should be replaced with the name (up to 4 characters) of the alternative reserved word table that has been created or NOOO if the IBM alternate table is desired.

Note: Effective with COBOL for z/OS and OS/390 3.1.0, the option WORD(NOOO) is no longer valid.

Processing Listings with Errors

Unpredictable results may occur if you postprocess a compiler listing with serious errors. If you use the preprocessor, you can avoid these potential problems. The preprocessor, in conjunction with the CONDDDIO command, internally checks the return code from the compiler and doesn't write out a DDIO listing that contains errors. Using the preprocessor prevents unnecessary processing of an error-filled compiler listing by not calling the postprocessor. For more information on the CONDDDIO command, refer to "CONDDDIO (CONDD)" on page 9-15

Language Processing Output Considerations

If you route your SYSPRINT or CWPPRTO output to a PDS or sequential dataset, please consider the following DCB restrictions.

- The preprocessor will force the SYSPRINT file to be LRECL=133 and RECFM=FBA.
- The postprocessor will force the CWPPRTO file to be LRECL=133 and RECFM=FBA.

Prevent Truncated Messages

Compiler messages are always truncated to 110 characters within the listing. To prevent this truncation and obtain the complete message, use the PROCESSOR($NO)

preprocessor option. Please refer to “PROCESSOR (PROC)” on page 9-28 for more information on this option.

CAUTION:

If you use a postprocessed source listing with XPEDITER/TSO, you may experience some loss of functionality. Refer to the *XPEDITER/TSO COBOL User’s Guide* for more information.

Handling Suppressed Statements

The postprocessor will not support listings that contain suppressed statements. If the listing contains suppressed statements, you must run the preprocessor. Suppressed statements include statements that are not generated in the compiler listing due to a COPY SUPPRESS for COBOL. If the postprocessor is run and it encounters any suppressed statements, you will receive the following message and the postprocessor will stop immediately.

```
PCDRVR051-S POSTPROCESSOR TERMINATED; PREPROCESSOR MUST BE USED DUE TO
COPY SUPPRESS
```

You can bypass this restriction by using the PROCESSOR($ASUP) parameter. This parameter allows a listing with suppressed statements to be postprocessed. Specify this parameter to postprocess listings with suppressed statements that have been stored in machine-readable format. $ASUP should be used with caution since there may be problems postprocessing a listing containing suppressed statements and using this listing with your Compuware products.

Effect of Optimization Logic on Execution Pointer

While the language processor supports the use of optimized code, occasionally the code execution pointer could follow a different path than expected due to the optimization logic. The effect of optimization on the COUNTS command may show unexpected results. This consideration only applies to XPEDITER/TSO and XPEDITER/CICS users.

Insufficient Primary Space Allocation

During preprocessing, you may receive a B37 abend on the CWPPRTI DD statement. This is due to insufficient space on the temporary volume used by MVS for the dynamically allocated dataset. To correct this problem, first try resubmitting the job. If the problem persists, you can include your own CWPPRTI DD statement in the preprocessor JCL. Be sure to allocate enough primary space to contain your compile listing.

Output Parameter Considerations

The OUTPUT parameter is the most commonly used COBOL language processor parameter. The OUTPUT parameter controls which type of listing (compiler or enhanced) is written to a DDIO file. This section discusses the DDIO and PRINT values for the parameter.

OUTPUT - DDIO Value

Use the DDIO value in the COBOL(OUTPUT) parameter or PROCESSOR(OUTPUT) parameter to specify which type of listing to write to the DDIO file. Only one type of listing can be written to a source listing file. The language processor uses the last-supplied parameter.

- If COBOL(OUTPUT(DDIO)) is specified, then the compiler listing is written to the DDIO file.
- If PROC(OUTPUT(DDIO)) is specified, then the enhanced listing is written to the DDIO file.

OUTPUT - PRINT Value

Use the PRINT value in the COBOL(OUTPUT) parameter or PROCESSOR(OUTPUT) parameter to specify which type of listing to print. One or both of these listings can be printed.

- If COBOL(OUTPUT(PRINT)) is specified, then the compiler listing is printed.
- If PROC(OUTPUT(PRINT)) is specified, then the enhanced listing is printed.

Performance Considerations

When using the COBOL command instead of the PROCESSOR command, the COBOL language processor uses less CPU time and performs fewer input/output operations.

Using the DDIO command to suppress the writing of certain sections of the compiler listing to a source listing file accomplishes the following:

- Performance is improved because the number of CPU cycles and input/output operations is reduced.
- Less space is used in the source listing file.

Usage Restrictions

The following restrictions apply to the COBOL language processor.

Restricted Compiler Options

- The COBOL language processor (all COBOL releases) does not support the compile option *CONTROL (*CBL) when the NOLIST, NOMAP, or NOSOURCE parameters for this option are specified.
- When using the LANGUAGE option, for all COBOL releases, do not place it on the CBL or PROCESS cards or the Compuware language processor cannot correctly parse the listing.

  If running the preprocessor, specify the LANGUAGE option in the preprocessor parameter list or in the CWPPRMO dataset.

  If running the postprocessor, specify the LANGUAGE option in the parameter list for the compiler.
- If using the NUM compile option, the number of program statements cannot exceed 999,999. This is due to a six-digit limitation on the program statement number.
- The preprocessor requires that the IGYCOPT (default options) module for COBOL/370 and VS COBOL II have ALOWCBL=YES specified. The preprocessor sets the required compiler options by adding CBL statements to the beginning of the source code.
- If the CA-OPTIMIZER III (for all releases) installation default is NOCOPT, you must specify PROCESSOR($NO) and LANGPARM(COPT) to execute the preprocessor properly.
- The COBOL compiler option FLAGSTD is not supported.
- If the postprocessor is used, the listing must be compiled complete with carriage control characters.

Other Restrictions

- ANS version 2 and 3 and FACOM COBOL compilers are not supported.
- Programs using segmentation or the LISTER feature are not supported.

- Compiler-generated data names, such as PRINT-SWITCH, -0002, or RPT.RCD used by the report writer feature, do not appear in the compiler listing.
- Use of the IBM Report Writer product (PO 5798-DYR) is supported. No other use of a compiler input or output exit routine is supported if it modifies the compiler input or output.
- COBOL preprocessor SYSIN and SYSLIB files must be fixed length 80-byte records.
- DBCS characters are not supported if the DBCS field contains mixed-case DBCS EBCDIC characters.
- User comments that simulate exactly the EXEC translator (CICS, SQL, or DB2) are not supported. For example, the following user comments are not distinguishable from a translated EXEC and may cause unpredictable results:

  ```
  *EXEC SQL ROLLBACK END-EXEC
  *****EXEC SQL ROLLBACK END-EXEC
  ```

  These types of comments may cause invalid enhanced FIND results and possible problems with GEN processing. These problems only occur during source listing viewing.
- ORACLE's SQL translator is not supported.
- The COPY SUPPRESS directive is not supported for the postprocessor.
- The batch option that produces multiple listings is not supported. Multiple compiler listings in SYSPRINT (read in as CWPPRTI) are also not supported.
- Optimized nested COBOL programs are supported; however, the CLIST information in the DDIO member may not be correct.

  The enhanced FIND feature is not supported for XPEDITER/TSO nested COBOL programs. Processing will continue; however, the enhanced FIND feature will be disabled.
- Declarative statements in COBOL/370 and VS COBOL II programs are not supported when OPTIMIZE is specified.
- Multiple data items on a single line are not supported when the MMAP or MDMAP compile option is used with CA-OPTIMIZER.
- COBOL keywords DIVISION and SECTION should not be split across multiple lines (continued from one line to the next).
- No object-oriented structures, keywords, or compiler options are allowed with the COBOL for MVS and VM compiler. In addition, the following reserved words are prohibited except when used in conjunction with the WORD(NOOO) option. See Using Alternative Reserved Word Tables on page 9-5 for more information.
  - CLASS-ID
  - END-INVOKE
  - INHERITS
  - INVOKE
  - LOCAL-STORAGE
  - METACLASS
  - METHOD
  - METHOD-ID
  - OBJECT
  - OVERRIDE
  - RECURSIVE
  - REPOSITORY
  - RETURNING
  - SELF
  - SUPER

Note: The last compiler supported for Abend-AID/COBOL ('CAID-TEXT') is AD/Cycle COBOL/370 1.1. Abend-AID/COBOL support has been functionally stabilized at Abend-AID Release 8.0. For further information, contact Abend-AID Technical Support.

Required Options for Supported COBOL Compilers

The following is a list of the COBOL compiler options that must be used with the COBOL language processor.

COBOL/370 1.1

- ALOWCBL (See note 3 on page 9-10)
- MAP
- NONUMBER (XPEDITER/TSO and XPEDITER/CICS) (See note 5 on page 9-10)
- NOTEST (XPEDITER/TSO and XPEDITER/CICS) (See note 5 on page 9-10)
- OFFSET or LIST (See note 1 on page 9-10)
- SOURCE
- XREF or XREF(SHORT) or XREF(FULL)

COBOL for MVS and VM 1.2 and 1.2.2 (MLE Support);
COBOL for OS/390 2.1 and 2.1.1 (MLE Support)
COBOL for OS/390 2.2
COBOL for z/OS and OS/390 3.1

- ALOWCBL (See note 3 on page 9-10)
- MAP
- NONUMBER (XPEDITER/TSO and XPEDITER/CICS) (See note 5 on page 9-10)
- NOTEST (XPEDITER/TSO and XPEDITER/CICS) (See note 5 on page 9-10)
- PGMNAME(COMPAT) (See note 6 on page 9-10)
- OFFSET or LIST (See note 1 on page 9-10)
- SOURCE
- XREF or XREF(SHORT) or XREF(FULL)

VS/COBOL II (all releases)

- ALOWCBL (See note 3 on page 9-10)
- MAP
- NOFDUMP (XPEDITER/TSO only) (See note 5 on page 9-10)
- NONUMBER (XPEDITER/TSO and XPEDITER/CICS) (See note 5 on page 9-10)
- NOTEST (XPEDITER/TSO and XPEDITER/CICS) (See note 5 on page 9-10)
- OFFSET or LIST (See note 1 on page 9-10)
- SOURCE
- XREF or XREF(SHORT) or XREF(FULL) (See note 2 on page 9-10)

OS/VS COBOL

- CLIST or PMAP
- DMAP
- NOBATCH
- NOCDECK
- NOFDECK
- NOFLOW (XPEDITER/TSO only) (See note 5 on page 9-10)
- NOLST
- NOSTATE (XPEDITER/TSO only) (See note 5 on page 9-10)
- NOSYMDMP
- NOTEST (XPEDITER/TSO and XPEDITER/CICS) (See note 5 on page 9-10)
- SOURCE
- VERB
- XREF or SXREF

CA-OPTIMIZER II (See note 4 on page 9-10)

- LIST or OFFSET or MOFFSET or CLIST or PMAP or MLIST
- MAP or DMAP or MMAP or MDMAP
- NOCXREF
- NOXMDMAP
- XREF or MXREF

CA-OPTIMIZER III (See note 4)

- CLIST or PMAP or MLIST (XPEDITER/TSO requires PMAP)
- DMAP or MDMAP
- NOCSL (XPEDITER/TSO only) (See note 5)
- NOCSXREF
- NOCXREF
- NOXCOUNT (XPEDITER/TSO only) (See note 5)
- NOXMDMAP

Notes:

1. Compuware recommends that you use the LIST option, instead of OFFSET, when using OPTIMIZE with COBOL/370 or VS/COBOL II. The parameter PROCESSOR($AOFF), is available for postprocessing the listing with the OFFSET compiler option. However, using this parameter may cause unpredictable results. For more information, see "$AOFF Parameter" on page 9-32.
2. When compiling with VS/COBOL II 1.2, use the XREF option.
3. This option is a COBOL installation option and cannot be overridden on the EXEC parameter. Please contact your systems programmer for more information.
4. When using CA-OPTIMIZER, all the options required of the COBOL compiler being used are also required by CA-OPTIMIZER.
5. These options are not forced on by the language preprocessor during the preliminary compile phase. See the PROCESSOR($NO) parameter, "PROCESSOR (PROC)" on page 9-28, for more information.
6. This option required only for releases earlier than CSS 7.8.

Enhanced Listing

The enhanced listing is a convenient source of quick reference information and program documentation. It significantly reduces the amount of documentation a programmer must examine during program development and maintenance. The enhanced listing is easier to work with because you do not have to make any hexadecimal calculations.

The enhanced listing merges DMAP information, BL, BLW, BLL, and Displacement into the Data Division. The enhanced listing displays subordinate data items with decimal lengths and relative positions (From - Thru). Due to space constraints, an asterisk (*) displays in the decimal length field if the Thru field length exceeds seven digits. OFFSET/CLIST information is also merged into the Procedure Division. CLIST information displays up to two verb addresses. The COBOL compiler options currently in effect are sorted alphabetically and appear at the beginning of the enhanced listing. The memory map, printed two-up, includes the displacement within the TGT as well as the displacement within the program.

The PROCESSOR(OUTPUT(PRINT)) command prints the enhanced listing (this is the default). You can also write the enhanced listing to a source listing file by specifying PROCESSOR(OUTPUT(DDIO)). See Figure 9-1, Figure 9-2, and Figure 9-3, for examples of enhanced listings.

Figure 9-1. Enhanced Listing - Merged DMAP/Data Division Map

```
BASE DISPL STMT    SEQ.      COBOL PROCESSOR  SUBPGM1    12.40.17 01/17/2002 PAGE     3    IDENT.          FROM THRU LENGTH
*                    ----+-*A-1-B--+----2----+----3----+----4----+----5----+----6----+----7-]--+----8

           00001   000001 IDENTIFICATION DIVISION.                                          00010000
           00002   000002 PROGRAM-ID.  'SUBPGM1'.                                           00020000
           00003   000003*INSTALLATION. COMPUWARE.                                          00030000
           00004   000004*REMARKS.                                                          00040000
           00005   000005*   THIS SUBROUTINE CALCS THE SAVINGS PLAN AMOUNT.                 00050000
           00006   000006*                                                                  00060000
           00007   000007 ENVIRONMENT DIVISION.                                             00070000
           00008   000008 DATA DIVISION.                                                    00080000
           00009   000009 WORKING-STORAGE SECTION.                                          00090000
BLW=1  000 00010   000010 01  CONSTANTS.                                                    00100000 CONSTANTS            9
BLW=1  000 00011   000011     03  MIN-FCTR         PIC S9V99     VALUE +0.02.              00110000            1    3    3
BLW=1  003 00012   000012     03  MAX-FCTR         PIC S9V99     VALUE +0.15.              00120000            4    6    3
BLW=1  006 00013   000013     03  ZERO-FCTR        PIC S9V99     VALUE  0.00.              00130000            7    9    3


**ERROR**  IKF2190I-W     PICTURE CLAUSE IS SIGNED, VALUE CLAUSE UNSIGNED. ASSUMED POSITIVE.

BLW=1  010 00014   000014 01  PRINT-VARIABLES.                                             00140000 PRINT-VARIABLES     11
BLW=1  010 00015   000015     03  PRINT-FCTR       PIC 9.99.                               00150000            1    4    4
BLW=1  014 00016   000016     03  MIN-MAX-PRT      PIC X(7)      VALUE SPACES.             00160000            5   11    7
BLW=1  020 00017   000017 01  WORK-VARIABLES.                                               00170000 WORK-VARIABLES      17
BLW=1  020 00018   000018     03  WS-GROSS-PAY     PIC 9(5)V99   VALUE 0                   00180000            1    7    7
BLW=1  027 00019   000019     03  WS-SAVINGS-FCTR  PIC 9V99      VALUE 0.                  00190000            8   10    3


**ERROR**  IKF1043I-W     END OF SENTENCE SHOULD PRECEDE 03 . ASSUMED PRESENT.
```

Figure 9-2. Enhanced Listing - File Definition and Record Layout for a VSAM File

```
BASE DISPL STMT   SEQ.     COBOL PROCESSOR  MAINPGM    17:25:59 01/17/2002 PAGE     4    IDENT.          FROM   THRU LENGTH
*                   ----+-*A-1-B--+----2----+----3----+----4----+----5----+----6----+----7-ž--+----8

           000023       DATA DIVISION.                                                      00230000
           000024       FILE SECTION.                                                       00240000
VSAM  FILE 000025       FD  TIMECARD                                                        00250000 ** FIXED LENGTH RECORDS **
           000026               RECORD CONTAINS 80 CHARACTERS                               00260000
           000027               DATA RECORD IS T-RECORD.                                    00270008
BLF=0  000 000028       01  T-RECORD               PIC X(80).                               00280000 T-RECORD                80
           000029       WORKING-STORAGE SECTION.                                            00290000
BLW=0  000 000030       01  TIME-RECORD.                                                    00300000 TIME-RECORD             80
BLW=0  000 000031           03  EMPLOYEE-NO        PIC 9(8).                                00310000            1      8      8
BLW=0  008 000032           03  MONTH-CODE         PIC XX.                                  00320000            9     10      2
BLW=0  00A 000033           03  EMPLOYEE-NAME      PIC X(32).                               00330000           11     42     32
BLW=0  02A 000034           03  GROSS-PAY          PIC 9(5)V99.                             00340000           43     49      7
BLW=0  031 000035           03  TOTAL-TAXES        PIC 9(4)V99.                             00350000           50     55      6
BLW=0  037 000036           03  SAVINGS-FACTOR     PIC V99.                                 00360020           56     57      2
BLW=0  039 000037           03  FILLER             PIC X(23).                               00370000           58     80     23
```

Figure 9-3. Enhanced Listing - Merged CLIST/OFFSET Information

```
STMT    SEQ                 COBOL PROCESSOR  SUBPGM1    12.40.17 01/17/2002 PAGE     4 VERB-1     ADDR-1 VERB-2     ADDR-2
        ----+-*A-1-B--+----2----+----3----+----4----+----5----+----6----+----7-]--+----8

00027          PROCEDURE DIVISION USING GROSS-PAY, SAVINGS-FCTR,                 00270000  ENTRY      000398
00028                                   SAVINGS-AMT, EMPLOYEE-NO.                00280000
00029          CALC-SAVINGS.                                                     00290000
00030              MOVE GROSS-PAY    TO WS-GROSS-PAY.                            00300000  MOVE       0003C0
00031              MOVE SAVINGS-FCTR TO WS-SAVINGS-FCTR.                         00310000  MOVE       0003CE
00032              MOVE ZEROS        TO WS-SAVINGS-AMT.                          00320000  MOVE       0003E0
00033              MOVE SPACES       TO MIN-MAX-PRT.                             00330000  MOVE       0003E6
00034              MULTIPLY WS-SAVINGS-FCTR BY WS-GROSS-PAY GIVING               00340000  MULTIPLY   0003F0
00035                 WS-SAVINGS-AMT.                                            00350000
00036              MOVE WS-SAVINGS-AMT TO SAVINGS-AMT.                           00360000  MOVE       000412
```

Reprocess Enhanced Listings Support

Beginning with CSS 7.8, an enhanced listing that was previously created by the COBOL Language Processor can now be reprocessed to create a DDIO source listing file member. Any enhanced listing created by prior releases of CSS is supported. Reprocessing enhanced listings can only be performed using the postprocessor. Use the same JCL to perform this function as you would to postprocess a COBOL listing, pointing the CWPPRTI DD statement to the dataset containing the enhanced listing.

Listing Features

The VS COBOL II enhanced listing shown in Figure 9-2 on page 9-11 displays the BLF and BLW cell and displacement that are printed on the left side of the corresponding source statement. The field name for each 77- and 01-level data item is printed on the right side of the corresponding source statement along with the total length of the item in decimal format. For data items subordinate to the 01-level, the enhanced listing displays the decimal "From Thru" position within the 01-level item and the total length of the item. These are printed to the right of the source statement.

If the data item has an OCCURS or REDEFINES clause, or is (or contains) the object of the DEPENDING ON option of an OCCURS clause, the appropriate indicator appears on the far right side of the source statement:

R	Redefines
O	Occurs
D/O	Depending on
F	File
E	External
G	Global

Record lengths, including the beginning and ending positions of all data elements within a record, are shown in decimal format (see Figure 9-2 on page 9-11). The location and length of all items in working storage are also shown in decimal format, eliminating the need to perform hexadecimal-to-decimal conversions. This format makes it very easy to correct program errors after compilation.

Procedure Division: The enhanced listing displays all verbs and displacements on the right side of the source statement as illustrated in Figure 9-3.

COBOL LP Command Descriptions and Syntax

This section describes the COBOL language processor commands and their parameters and values.

CICSTEST (CIC)

The CICSTEST command instructs the COBOL language processor to print either a warning or severe message regarding the status of the COBOL compile options for running a program in CICS.

The following diagram illustrates the syntax for the CICSTEST command.

```
                 ┌(NOOptions)──────────┐
►►─CICstest──────┼─────────────────────┼──────────►◄
                 └(Options(┌Severe──┐))┘
                           ├Warning─┤
                           └No──────┘
```

NOOPTIONS (NOO):

(Default) Does not print either a warning or severe error message, even if the program was compiled with invalid options and/or verbs for CICS.

OPTIONS (O):

Specifies the options for controlling the CICS test error messages. Select from the following parameter values:

SEVERE (S):

Prints a warning and a severe message if the program was compiled with invalid CICS options.

WARNING (W):

Prints a warning message if the program was compiled with invalid CICS options.

NO (N):

Does not print a message.

COBOL (COB)

The COBOL command controls the compiler listing destination. The parameter for the COBOL command is OUTPUT.

The following diagram illustrates the syntax for the COBOL command OUTPUT parameter.

```
                    ┌NOPRint─┐    ┌NODDio─┐
►►─COBol(OUtput(─┬─┼────────┼─,──┼───────┼─┬─))──────►◄
                 │ └PRint───┘    └DDio───┘ │
                 ├BOth─────────────────────┤
                 └NONe─────────────────────┘
```

OUTPUT (OU):

Specifies whether the compiler listing will be printed or written to a DDIO file. COBOL(OUTPUT(DDIO)) and PROCESSOR(OUTPUT(DDIO)) are mutually exclusive. The language processor uses the last-supplied parameter. Select from the following values:

NOPRINT (NOPR):

(Default) Does not print the compiler listing.

PRINT (PR):

Prints the compiler listing.

NODDIO (NODD):

(Default) Does not write the compiler listing to a DDIO file.

DDIO (DD):

Writes the compiler listing to a DDIO file.

BOTH (BO):

Has the same effect as specifying COBOL(OUTPUT(PRINT,DDIO)).

NONE (NON):

Does not print the compiler listing and does not write it to a DDIO file.

CONDDDIO (CONDD)

The CONDDDIO command specifies that the source listing will be written to the source listing file unless the compiler return code exceeds **x**. The default is 4. Permissible values are 0-8.

Note: Unpredictable results may occur when attempting to use a listing created with the CONDDDIO parameter set to 8 and the compiler return code set to 8. The listing will be stored in the DDIO file with these settings; however, unpredictable results may occur.

The CONDDDIO command can only be used with the preprocessor.

The following diagram illustrates the syntax for the CONDDDIO command.

```
►►─CONDDdio(─┬─4─┬─)──────►◄
             └─x─┘
```

CONDERRL (CONDE)

The CONDERRL command, valid only for OS/VS COBOL and CA-OPTIMIZER, indicates that the hard copy listing will contain only error messages and lines in error if the compiler condition code is greater than **x**. If the compiler condition code does not exceed **x**, then all requested portions of the listing will be printed. If **x** is omitted, the value defaults to **4** for OS/VS COBOL and **6** for CA-OPTIMIZER.

The CONDERRL command can be used only with the preprocessor.

The following diagram illustrates the syntax for the CONDERRL command for OS/VS COBOL.

```
►►─CONDErrl(─┬─4─┬─)──────►◄
             └─x─┘
```

The following diagram illustrates the syntax for the CONDERRL command for CA-OPTIMIZER.

```
►►─CONDErrl(─┬─6─┬─)──────►◄
             └─x─┘
```

CONDSTEP (CONDS)

The CONDSTEP command controls the condition code set by the postprocessor when it is invoked by the preprocessor. The CONDSTEP parameter is ignored when the postprocessor is invoked as a separate job step after the compile; the postprocessor return code is always used.

The CONDSTEP command can be used only with the preprocessor.

The following diagram illustrates the syntax for the CONDSTEP command.

```
                 ┌HIghest─┐
►►─CONDStep(─┼────────┼─)──────►◄
                 └COmpiler─┘
```

HIGHEST (HI):

(Default) The condition code will be set to the value of the COBOL language processor return code or the compiler return code, whichever is higher.

COMPILER (CO):

The condition code is always set to the compiler return code. This option is typically used to allow a subsequent linkedit step to execute when the compile executes normally but the postprocessor is unable to execute successfully.

DDIO (DD)

The DDIO command identifies the sections of the COBOL compiler listing that are written to the source listing file. The parameter for the DDIO command is OUTPUT.

The following diagram illustrates the syntax for the DDIO command OUTPUT parameter.

```
►►─DDio(OUtput(─┬─NOFInd─┬─,─┬─NOCOmpress─┬─,─┬─XRef───────┬─,─┬─PMap───────┬─,──►
                └─FInd───┘   └─COmpress────┘   └─NOXRef────┘   └─NOPMap────┘

►─┬─DMap───────┬─,─┬─LIst───────┬─,─┬─CList───────┬─,─┬─OFfset──────┬─))─►◄
  └─NODMap─────┘   └─NOLIst─────┘   └─NOCList─────┘   └─NOOFfset────┘
```

OUTPUT (OU):

Specifies the options for a source listing that is written to the source listing file.

NOFIND (NOFI):

(Default) Does not generate records for enhanced FIND.

FIND (FI):

Generates records needed for the enhanced FIND feature.

Note: XPEDITER/TSO (only) requires the FIND command in order to use the enhanced FIND feature.

NOCOMPRESS (NOCO):

(Default) Does not compress the compiler listing. The size of the LP source members is not reduced.

COMPRESS (CO):

Enables compression. Depending on the size of your program and the options you set, the size of LP source members may be reduced.

Note: Performance is affected when compression is used. More CPU and memory is required for compression/decompression of the compiler listing. This is especially important for online usage.

XREF (XR):

(Default) Writes the XREF sections of the listing.

NOXREF (NOXR):

Does not write the XREF sections of the listing.

PMAP (PM):

(Default) Writes the PMAP section of the listing.

NOPMAP (NOPM):

Does not write the PMAP section of the listing.

DMAP (DM):

(Default) Writes the DMAP section of the listing.

NODMAP (NODM):

Does not write the DMAP section of the listing.

LIST (LI):

(Default) Writes the Assembler section of the listing.

NOLIST (NOLI):

Does not write the Assembler section of the listing.

CLIST (CL):

(Default) Writes the CLIST section of the listing.

NOCLIST (NOCL):

Does not write the CLIST section of the listing.

OFFSET (OF):

(Default) Writes the OFFSET section of the listing.

NOOFFSET (NOOF):

Does not write the OFFSET section of the listing.

File-Overrides Commands

The File-overrides commands are used to override the default file definitions. The following overrides are available:

- WRK0 - WRK4
- PRTI
- PRTO
- ERRM
- LOAD
- DECK

The File-overrides commands can be used only with the postprocessor.

Note: The LOAD or DECK ddname is used by the COBOL language processor depending on the options selected for the compile step. If both LOAD and DECK are specified, both will be processed by the COBOL language processor. If you are using a temporary dataset for your object, and only one option (LOAD or DECK) is specified, eliminate the ddname that corresponds to the option that is NOT specified. For example, if you specify LOAD, then eliminate the DECK ddname.

WRK0-WRK4 File-Overrides Commands

Specifies the work files.

```
>>-WRKn(1)(┬──────────────────────────────────────┬)──>◄
           └─DD(ddname),BLK(blksize),BUF(buffers)─┘
```

Note (1): In this diagram, WRKn represents WRK0 - WRK4.

If you omit DD(ddname), the assumed ddname will be CWPWRKn.

PRTI File-Overrides Command

Specifies the compiler listing input.

```
>>-PRTI(┬──────────────────────────────────────────────────────────┬)──>◄
        └─DD(ddname),BLK(blksize),BUF(buffers),LRECL(lrecl),RECFM(recfm)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPPRTI.

PRTO File-Overrides Command

Specifies COBOL language processor output.

```
>>-PRTO(┬──────────────────────────────────────────────────┬)──>◄
        └─DD(ddname),BLK(blksize),BUF(buffers),LRECL(lrecl)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPPRTO.

ERRM File-Overrides Command

Prints the COBOL language processor error messages.

```
►►─ERRM(┬──────────────────────────────────────┬)──►◄
        └DD(ddname),BLK(blksize),BUF(buffers)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPERRM.

LOAD File-Overrides Command

Specifies the object library (indicated by the LOAD option of the compiler).

```
►►─LOAD(┬──────────────────────────────────────┬)──►◄
        └DD(ddname),BLK(blksize),BUF(buffers)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPLOAD.

DECK File-Overrides Command

Specifies the object library (indicated by the DECK option of the compiler).

```
►►─DECK(┬──────────────────────────────────────┬)──►◄
        └DD(ddname),BLK(blksize),BUF(buffers)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPDECK.

Optional Override Values

The keywords below cause the language processor to override its default values:

DDNAME (DD):

Specify a ddname instead of accepting the default ddname.

BLKSIZE (BLK):

Specify a decimal blocksize instead of accepting the default blocksize.

BUFFERS (BUF):

Specify a decimal number as the number of buffers instead of accepting the default number of buffers, which is **2**. Only the first two buffers are used for buffering input/output. Extra buffers are used as in-core work space for the file. Therefore, records placed in the work space will never be written to the work files. If the storage is available, use the BUFFERS parameter with the BLKSIZE parameter to increase the language processor speed.

If the language processor is running in the MVS/XA or MVS/ESA environments, and is above the 16-MB line, a value of zero specifies that as much storage as is available will be used. Using zero places a greater amount of overhead on startup and should not be used for small programs.

LRECL (LR):

Specify a decimal number as the logical record length instead of accepting the default logical record size.

RECFM (RE):

Specify a record format instead of accepting the default record format. Valid formats include:

F	Fixed
FA	FixedASA
FM	FixedMachine
V	Variable
VA	VariableASA
VM	VariableMachine

INCLUDE (IN)

Use the INCLUDE command to include a library or sequential file member that contains another set of language processor options to be processed.

An included set of options can, in turn, include another set of options. If a recursive chain of INCLUDEs is encountered, the language processor will stop with an error message.

Duplicate or conflicting parameters are resolved as follows:

- Options are processed left to right.
- When an INCLUDE is encountered, the included member is processed immediately. Options following the INCLUDE will be processed after the included member is processed.
- The last option processed takes precedence.
- The sources of options are, in order of descending priority:
  - forced options in the COBOL language compiler installation default module
  - CBL options in the source file (or equivalent for other compilers)
  - PARM options on the EXEC card
  - options specified in the CWPPRMO dataset
  - default options in the COBOL language compiler installation default module.

The INCLUDE command can be used only with the preprocessor.

The following diagram illustrates the syntax for the INCLUDE command.

```
                 ┌CWPPRMO┐
►►─INclude(─┼───────┼─,─┬───────┬─)──►◄
                 └ddname─┘   └member─┘
```

ddname:

Specifies the DDNAME of the JCL statement that specifies the dataset name. If a member is not specified, then the ddname must point to a sequential file or library member.

member:

Specifies the library member name. If the ddname is not specified, then the default CWPPRMO will be used and a comma must precede the member name.

LANGPARM (LANGP)

The LANGPARM command specifies parameters to be passed to the compiler invoked by the language processor. If the language processor is run as a separate job step, this option is ignored.

If you use the CWPPRMO dataset to specify your compiler options, use the LANGPARM command so that the preprocessor can distinguish between compiler options and language processor options with the same name. The LANGPARM command allows the preprocessor to identify compiler options that conflict with language processor names, thus preventing unpredictable results. If you use the PARM statement to specify your compiler options, you can also use the LANGPARM command.

The LANGPARM command can be used only with the preprocessor.

The following diagram illustrates the syntax for the LANGPARM command.

```
>>-LANGParm(┬──────────────┬)──────>◄
            └parm1...parmn─┘
```

parm:

Specifies a parameter to be passed.

LANGUAGE (LANG)

The LANGUAGE command specifies the type of compiler for the COBOL language preprocessor and postprocessor.

For the preprocessor, the LANGUAGE command specifies which compiler to execute.

Note: The preprocessor accepts any of the COBOL language parameters as being equivalent because the compiler program name (IGYCRCTL) is the same for all compilers. The compiler that is executed depends on the library specified in the STEPLIB, JOBLIB, and/or LINKLIST concatenation.

For the postprocessor, the LANGUAGE command identifies the compiler version that produced the compiler listing. This is useful when the compiler listing does not show a COBOL header (usually the first line of a listing) that specifies the compiler version.

The following diagram illustrates the syntax for the LANGUAGE command.

```
                 ┌VSCOBOLII──────┐
►►──LANGuage(─┼───────────────┼─,─┬──────────────┬─)──────►◄
                 ├VSCOBOL───────┤   └CA-OPTIMIZER─┘
                 ├VSCOBOLIIREL3─┤
                 ├VSCOBOLIIREL4─┤
                 ├COBOL/MVS─────┤
                 ├COBOL/370─────┤
                 ├COBOL/390─────┤
                 └COBOLZ/OS─────┘
```

The parameters you may select from are:

VSCOBOLII:

(Default) The language processor assumes a VS COBOL II listing if another compiler name cannot be determined.

VSCOBOL:

The language processor assumes an OS/VS COBOL I listing if the compiler cannot be determined.

VSCOBOLIIREL3:

The language processor assumes a VS COBOL II Release 3 listing if the compiler cannot be determined.

VSCOBOLIIREL4:

The language processor assumes a VS COBOL II Release 4 listing if the compiler cannot be determined.

COBOL/MVS:

The language processor assumes a COBOL for MVS and VM listing if the compiler cannot be determined.

COBOL/370:

The language processor assumes a COBOL/370 listing if the compiler cannot be determined.

COBOL/390:

The language processor assumes a COBOL for OS/390 listing if the compiler cannot be determined.

COBOLZ/OS

The language processor assumes a COBOLZ/OS listing if the compiler cannot be determined.

CA-OPTIMIZER:

The language processor will use the CA-OPTIMIZER tables for the version of COBOL that processed the listing. CA-OPTIMIZER can be specified with VSCOBOL or VSCOBOLII. If CA-OPTIMIZER is specified with VSCOBOL, then CA-OPTIMIZER III is invoked. If CA-OPTIMIZER is specified with VSCOBOLII, then CA-OPTIMIZER II is invoked.

PRINT (PR)

The PRINT command identifies the sections of the listing that will be printed. The parameter for the PRINT command is OUTPUT.

The following diagram illustrates the syntax for the PRINT command OUTPUT parameter.

```
                    ┌LIst──┐   ┌CList──┐   ┌OFfset──┐   ┌XRef──┐
►►─PRint(OUtput(─┼──────┼─,─┼───────┼─,─┼────────┼─,─┼──────┼─,──►
                    └NOLIst┘   └NOCList┘   └NOOFfset┘   └NOXRef┘

   ┌DMap──┐   ┌PMap──┐   ┌MAp──┐   ┌SOurce──┐
►─┼──────┼─,─┼──────┼─,─┼─────┼─,─┼────────┼─))──►◄
   └NODMap┘   └NOPMap┘   └NOMAp┘   └NOSOurce┘
```

OUTPUT (OU):

Specifies the options for the listing that will be printed.

LIST (LI):

(Default) Prints the Pseudo-Assembler section of the listing.

NOLIST (NOLI):

Does not print the Pseudo-Assembler section of the listing.

CLIST (CL):

(Default) Writes the CLIST section of the listing.

NOCLIST (NOCL):

Does not write the CLIST section of the listing.

OFFSET (OF):

(Default) Prints the Offset section of the listing.

NOOFFSET (NOOF):

Does not print the Offset section of the listing.

XREF (XR):

(Default) Prints the XREF sections of the listing.

NOXREF (NOXR):

Does not print the XREF sections of the listing.

DMAP (DM):

(Default) Prints the DMAP section of the listing for VS COBOL.

NODMAP (NODM):

Does not print the DMAP section of the listing for VS COBOL.

PMAP (PM):

(Default) Writes the PMAP section of the listing.

NOPMAP (NOPM):

Does not write the PMAP section of the listing.

MAP (MA):

(Default) Prints the MAP section of the listing for VS COBOL II.

NOMAP (NOMA):

Does not print the MAP section of the listing for VS COBOL II.

SOURCE (SO):

(Default) Prints the Source section of the listing.

NOSOURCE (NOSO):

Does not print the Source section of the listing.

PROCESSOR (PROC)

The PROCESSOR command specifies the output parameters of the COBOL language processor. The PROCESSOR command parameters are:

- BYPASS
- ERRORS
- OUTPUT
- POSITION and OFFSET (mutually exclusive)
- SCALELINE
- STATISTICS
- TEXT
- WARNINGS
- $AOFF
- $ASUP
- $NG
- $NIDR
- $NO

BYPASS (BY) Parameter

Specifies whether to run the preprocessor and compiler without running the postprocessor. The following diagram illustrates the syntax for the BYPASS parameter.

```
►►─PROCessor(─┬─NOBYpass─┬─)──►◄
              └─BYpass───┘
```

NOBYPASS (NOBY):

(Default) Runs the postprocessor.

BYPASS (BY):

Does not run the postprocessor.

ERRORS (ER) Parameter

Specifies how the language processor error messages will be printed. The following diagram illustrates the syntax for the ERRORS parameter.

```
►►─PROCessor(ERrors(─┬─UPper-case─┬─))──►◄
                     └─MIxed-case─┘
```

UPPER-CASE (UP):

(Default) Errors will print in uppercase letters.

MIXED-CASE (MI):

Errors will print in uppercase and lowercase letters.

OUTPUT (OU) Parameter

Defines options for printing the enhanced listing or writing it to a source listing file. The COBOL(OUTPUT(DDIO)) and PROCESSOR(OUTPUT(DDIO)) are mutually exclusive. The language processor uses the last-supplied parameter.

The following diagram illustrates the syntax for the OUTPUT parameter.

```
                          ┌PRint──┐     ┌NODDio─┐
▶▶─PROCessor(OUtput(──┬──┼───────┼──,──┼───────┼──┬──))──────▶◀
                      │  └NOPrint─┘     └DDio───┘  │
                      ├BOth──────────────────────────┤
                      └NONe──────────────────────────┘
```

PRINT (PR):

(Default) Prints the enhanced listing.

NOPRINT (NOPR):

Does not print the enhanced listing.

NODDIO (NODD):

(Default) Does not write the enhanced listing to a source listing file.

DDIO (DD):

Writes the enhanced listing to a source listing file.

BOTH (BO):

Has the same effect as specifying PROCESSOR(OUTPUT(PRINT,DDIO)).

NONE (NON):

Does not print the enhanced listing and does not write it to a source listing file.

POSITION (POS) Parameter

Specifies whether positional data will be used in the From-Through data of the enhanced listing Data Division section. The POSITION and OFFSET parameters are mutually exclusive.

The following diagram illustrates the syntax for the POSITION parameter.

```
                  ┌POSition──┐
▶▶─PROCessor(─┼──────────┼─)──────▶◀
                  └NOPOSition─┘
```

POSITION (POS):

(Default) Positional data will be used.

NOPOSITION (NOPOS):

Offset data will be used.

OFFSET (OFF) Parameter

Specifies whether offset data will be used in the From-Through data of the enhanced listing Data Division section. The POSITION and OFFSET parameters are mutually exclusive.

The following diagram illustrates the syntax for the OFFSET parameter.

```
                 ┌OFFset──┐
▶▶─PROCessor(─┼────────┼─)──▶◀
                 └NOOFFset─┘
```

OFFSET (OFF):

(Default) Offset data will be used.

NOOFFSET (NOOFF):

Positional data will be used.

SCALELINE (SCA) Parameter

Specifies whether a scale line will be added to the enhanced listing. The following diagram illustrates the syntax for the SCALELINE parameter.

```
                 ┌SCAleline──┐
▶▶─PROCessor(─┼───────────┼─)──▶◀
                 └NOSCAleline─┘
```

SCALELINE (SCA):

(Default) A scale line will be added.

NOSCALELINE (NOSCA):

A scale line will not be added.

STATISTICS (ST) Parameter

Specifies whether to include statistics in the enhanced listing that compare the number of lines and pages the COBOL language processor printed to the number of lines and pages of the compiler listing. These statistics provide an effective means of showing the savings achieved by reformatting the compiler listing into the enhanced listing.

The following diagram illustrates the syntax for the STATISTICS parameter.

```
                 ┌STatistics──┐
▶▶─PROCessor(─┼───────────┼─)──▶◀
                 └NOSTatistics─┘
```

STATISTICS (ST):

(Default) Prints all the statistics.

NOSTATISTICS (NOST):

Suppresses the printing of certain statistics.

Note: To print all of the statistics, you must also use the TEXT parameter. If all statistics are requested without using the TEXT parameter, the net effect will be that of specifying NOSTATISTICS; i.e., certain statistics will be omitted.

TEXT (TE) Parameter

Specifies when text is generated by the language processor. The TEXT parameter causes the COBOL language processor to enhance the object module by appending non-executable text code to it. The text code generated by the TEXT parameter is used only by Abend-AID to provide Abend-AID/COBOL support.

The following diagram illustrates the syntax for the TEXT parameter.

```
                    ┌ALl──────┐
▶▶─PROCessor(TExt(─┼─────────┼─))──────▶◀
                    └NONe─────┘
```

ALL (AL):

(Default) Generates text for the COBOL program. If you use the postprocessor when TEXT=ALL, you must specify either CWPLOAD or CWPDECK as input to the postprocessor step.

NONE (NON):

Does not generate any text.

Note: For COBOL 390 and later releases of COBOL the default is TEXT(NONE).

See the note in “Usage Restrictions” on page 9-7 for more information on using the COBOL for MVS and VM compiler.

WARNINGS (WA) Parameter

Specifies where warning messages are placed in the enhanced listing. The following diagram illustrates the syntax for the WARNINGS parameter.

```
                    ┌BOth──────────────────────────┐
▶▶─PROCessor(WArnings(─┼──┬──────────┬──,──┬──────────┬─))──▶◀
                    │  ├FRont──┤     ├EMbed──┤  │
                    │  └NOFRont─┘     └NOEMbed┘  │
                    └NONe──────────────────────────┘
```

BOTH (BO):

(Default) Prints and embeds warning messages.

NONE (NON):

Does not print or embed warning messages.

FRONT (FR):

Prints warnings in the front of the listing.

NOFRONT (NOFR):

Does not print warnings.

EMBED (EM):

Embeds warnings after the appropriate statement.

NOEMBED (NOEM):

Does not embed warnings.

$AOFF Parameter

Allows postprocessing of a listing with the OFFSET compiler option when using OPTIMIZE. If the $AOFF parameter is specified, you will receive a warning message indicating that the OFFSET option was used instead of LIST. $AOFF may cause unexpected results when using XPEDITER/TSO or XPEDITER/CICS to debug optimized code.

The following diagram illustrates the syntax for the $AOFF parameter.

```
►►─PROCessor(┬─────────┬)──►◄
             └$AOFF────┘
```

$ASUP Parameter

If $ASUP is specified, the postprocessor will allow suppressed statements to be present in the listing. For more information on $ASUP, refer to “Handling Suppressed Statements” on page 9-6.

The following diagram illustrates the syntax for the $ASUP parameter.

```
►►─PROCessor(┬─────────┬)──►◄
             └$ASUP────┘
```

$NG Parameter

Specifies that the preprocessor’s support of COPY SUPPRESS will be bypassed. The following diagram illustrates the syntax for the $NG parameter.

```
►►─PROCessor(┬─────────┬)──►◄
             └$NG──────┘
```

Note: The $NG option should not be specified if the WORD option is used for alternative reserved words.

$NIDR Parameter

Specifies that placement of the date and time stamp on the object module will be bypassed. The following diagram illustrates the syntax for the $NIDR parameter.

```
>>─PROCessor(─┬──────┬─)──────>◄
              └$NIDR─┘
```

$NO Parameter

Specifies that the preliminary compilation performed by the preprocessor will be bypassed. The preliminary compilation is normally used to determine the compiler options currently in effect. However, if you know that your options satisfy the requirements of the language processor, you can bypass the preliminary compilation to reduce the preprocessor's execution time.

The following diagram illustrates the syntax for the $NO parameter.

```
>>─PROCessor(─┬──────┬─)──────>◄
              └$NO─┘
```

Note: The $NO option should not be specified if the WORD option is used for alternative reserved words.

PROGRAM-NAME (PRO)

The PROGRAM-NAME command specifies an override name for members in the source listing file.

The following diagram illustrates the syntax for the PROGRAM-NAME command.

```
>>-PROgram-name(-+-------+-)-------><
                 '-name-'
```

name:

Specifies an alternative name for a source listing written to the source listing file.

SYSTEM (SYS)

The SYSTEM command performs miscellaneous functions. The following diagram illustrates the syntax for the SYSTEM command.

```
>>-SYStem(-+-------+,-+-------+,-+-------+,-+-------+,-+-------+)-><
           '-ESORT-'  '-TRace-'  '-TIme-'  '-PTF-'  '-ZAP-'
```

ESORT:

Specifies that an estimated sort count will be used instead of an actual sort count. The language processor keeps track of how many records should be fed into the sort phase. If a discrepancy exists, a sort error will occur and the language processor stops. This option allows the language processor to continue if a discrepancy exists.

CAUTION

Extreme caution should be used when specifying ESORT. Although the language processor will complete, the DDIO member may be unusable. ESORT should only be used under the direction of Compuware Technical Support.

TRACE (TR):

Specifies that the internal trace will be printed to CWPERRM.

TIME (TI):

Specifies that the internal time will be printed with the trace.

PTF:

Specifies that the PTF numbers currently installed on the CSS library will be printed. PTF is used by Compuware Technical Support during problem diagnosis.

ZAP:

Specifies that the PTF numbers currently installed on the CSS library will be printed. ZAP is used by Compuware Technical Support during problem diagnosis.

Note: The ZAP parameter is allowed for compatibility with pre-SMP/E releases.

Chapter 10.
PL/I Language Processor

IMPORTANT

> **This chapter, when referring to the PL/I language processor, also refers to the VisualAge PL/I and Enterprise PL/I language processors.**

A language processor writes a source listing to a VSAM or sequential DDIO file. For PL/I, the source listing includes a compiler listing and other information about a program. This chapter describes the outputs and commands associated with the PL/I language processor.

Operation

The PL/I language processor reads the compiler listing produced by the PL/I compiler and writes the compiler listing to the DDIO file and SYSPRINT. The source listing can then be used to provide source support during a debugging or diagnostic session using any of the following products:

- Abend-AID
- Abend-AID for CICS
- XPEDITER/CICS
- XPEDITER/TSO.

The PL/I language processor can be run as either a preprocessor or postprocessor. Compuware recommends that you use the preprocessor. The postprocessor should only be used in situations where compiled source listings have been stored in machine-readable format, and where the program did not use the %NOPRINT statement.

Preprocessor

Some information about a program module is not always available from the compiler listing. The preprocessor, in addition to gathering information from the compiler listing, gathers information from SYSIN and SYSLIB. The preprocessor provides additional functionality for XPEDITER/CICS and XPEDITER/TSO, and is required for PL/I programs that use the %NOPRINT statement.

Preprocessor Steps

1. Determines the proper compiler options required to process the listing file.
2. Automatically invokes the compiler or assembler to compile or assemble your source program.
3. Writes the listing to the DDIO file and to SYSPRINT.

Preprocessor Benefits

- **Better handling of compiler errors**. By using the preprocessor, you can avoid the potential problems of processing a listing with compiler errors. The preprocessor, in conjunction with the CONDDDIO parameter, internally checks the return code from the compiler and doesn't write a DDIO listing that contains errors. For example, if you set the CONDDDIO parameter to 8, the preprocessor will write the compiler listing to the DDIO file unless the compiler return code exceeds 8. The preprocessor can process the compiler errors more effectively than the postprocessor.

- **Capturing of suppressed source code**. When the following parameter is used, sections of source code can be suppressed from the compiler listing:

  For PL/I %NOPRINT

  The preprocessor captures this information from SYSIN and SYSLIB.

- **Automated compiler options**. The postprocessor requires that certain compiler options be specified in order to process all needed sections of the compiler listing. The preprocessor can automatically pass the required options to the compiler.

  For a list of the required compiler options, refer to the "Required Options for Supported PL/I Compilers" on page 10-10.

- **Simplified JCL**. While the postprocessor requires that the user add a step after the compile step, the preprocessor requires only that minor modifications be made to your existing compile step.

Dynamically Allocated Files

The following files are dynamically allocated by the Compuware preprocessor. The attributes can be overridden by specifying a DD in the JCL.

Table 10-1. Files Dynamically Allocated by the Preprocessor

DDname	Attributes
CWPWRKn (0-6)	BLKSIZE=19000 SPACE(TRK,(100,80)) UNIT=SYSDA
SORTWK01	BLKSIZE=19000 SPACE(TRK,(100,80)) UNIT=SYSDA
TEMPLIN	BLKSIZE=3200 SPACE(TRK,(200,100)) UNIT=SYSDA
CWPERRM	SYSOUT=*
SYSPRINT	SYSOUT=*
SYSOUT	SYSOUT=*
CWPPRTI	BLKSIZE=19000 SPACE(TRK,(100,200)) UNIT=SYSDA
CWPPDS2	SPACE (TRK,(100,80,80)) UNIT=SYSDA

CWPPRTO is dynamically allocated as a work file (for the preprocessor). SYSPRINT is where the listing goes.

Postprocessor

The postprocessor is executed as a single step — can be a separate job. It reads in the listing created from the compiler. Information is gathered from the source listing, XREF, data maps, and object code sections of the listing. This information is stored in a DDIO file member and is used by various Compuware products.

While the preprocessor is the preferred method of loading the compiled listing to the DDIO, there are situations that necessitate use of the postprocessor as a viable alternative. This is commonly found in production environments where the listings may be archived. In order for the listings to populate the DDIO, they must be compiled using the required options specified in "Required Options for Supported PL/I Compilers" on page 10-10. Note that these are usually the default options at many shops.

If the programs contain suppressed source code, certain additional control statements must be added to the language processor.

Benefits of Postprocessing

- Exact match of compiled output listings with the executable load module. This is especially important in production environments where time is critical.

- Eliminates the risk of copybook or source code changes prior to recompiling the code since the listing reflects the code that is being executed.
- Significantly less time is needed to process source listings as opposed to recompiling the source code.

Summary of PL/I Language Processor Commands

The following is a list and brief description of the commands available for the PL/I language processor. Each command is discussed in detail later in this chapter.

CONDDDIO	Specifies that the source listing will be written to the source listing file unless the compiler return code exceeds the maximum value you set.
CONDSTEP	***(Can only be used with the preprocessor)*** Controls the condition code set by the postprocessor when it is invoked by the preprocessor.
DDIO	Identifies the sections of the PL/I compiler listing that are written to the source listing file.
File-overrides	***(Can only be used with the postprocessor)*** Used to override the default file definitions.
INCLUDE	***(Can only be used with the preprocessor)*** Used to include a library or sequential file member that contains another set of language processor options to be processed.
LANGPARM	***(Can only be used with the preprocessor)*** Specifies parameters to be passed to the compiler invoked by the language processor.
LANGUAGE	Specifies the type of compiler for the PL/I language preprocessor and postprocessor.
PL/I	Controls the compiler listing destination.
PRINT	Identifies the listing sections that will be printed.
PROCESSOR	Specifies the output parameters of the PL/I language processor.
PROGRAM-NAME	Specifies an override name for members in the source listing file.
SYSTEM	Performs miscellaneous functions

The commands can be entered in two ways:

- As parameters on the PARM statement of the EXEC — (Overrides input from the CWPPRMO file.)
- As input from the CWPPRMO file — (Comments are not valid and all command strings must be within columns 1 through 71.)

Each command has parameters (and some parameters have values) that must be written immediately after the commands, be enclosed in parentheses, and separated by commas.

If conflicting command options are entered, the last option specified is used.

Usage Considerations

This section includes information that you need to know in order to be able to use the PL/I language processor most efficiently.

Language Processor DD Statements

The following table lists the DD statements for the PL/I language processor.

Table 10-2. Language Processor DD Statements.

DDname	Purpose
CWPWRK (0-6)	LP work files
CWPPRMO	LP parameter input dataset
CWPDDIO	Target source listing DDIO file
CWPPRTI	Compiler listing input to postprocessor
CWPPRTO	Compiler listing output from postprocessor
CWPLOAD	Option LOAD or OBJECT module input to postprocessor
CWPDECK	Option DECK module input to postprocessor
CWPERRM	LP Error Messages

Note: CWPLOAD and CWPDECK correspond to the SYSLIN and SYSPUNCH compiler outputs. If you specify compiler option OBJECT, you should use CWPLOAD in your postprocessor JCL. If you specify compiler option DECK, you should use CWPDECK in your postprocessor JCL. Do not feed the SYSLIN output into CWPDECK or the SYSPUNCH output into CWPLOAD. The updated OBJECT contained in CWPLOAD and CWPDECK should be used in the program's linkedit step. The preprocessor does not use CWPLOAD and CWPDECK.

PL/I Samples and Program Names

The following is a list of members in the CSS installation library that contain sample JCL for executing the PL/I language processor:

Table 10-3. PL/I Sample JCL Members

CXPLIPRE	JCL for running the PL/I language preprocessor
CXPLI	JCL for running the PL/I language postprocessor
CXPLI2	JCL for running the PL/I language postprocessor with compiler listings stored in machine-readable format
CXLPPLI	Sample PL/I language processor options

The following program names are used for the PL/I language processor:

Table 10-4. PL/I Program Names

CWPPMAIN	PL/I preprocessor
CWPPDRVR	PL/I postprocessor

Other Usage Considerations

This section describes special considerations for using the PL/I language processor. Certain restrictions also apply to the PL/I language processor in the following areas:

- Declaration of variables
- Compiler/run-time options
- Programming techniques

For a complete list of these restrictions, see “Usage Restrictions and Examples”.

Processing Listings with Errors

Unpredictable results may occur when you postprocess a compiler listing with serious errors. If you use the preprocessor, however, you can avoid these potential problems. The preprocessor, in conjunction with the CONDDDIO parameter, internally checks the return code from the compiler and doesn’t process a listing that contains errors. Using the preprocessor prevents unnecessary processing of an error-filled compiler listing by not calling the postprocessor. For more information on the CONDDDIO parameter, refer to “CONDDDIO (CONDD)” on page 10-11.

Insufficient Primary Space Allocation

During preprocessing, you may receive a B37 abend on the CWPPRTI DD statement. This is due to insufficient space on the temporary volume used by MVS for the dynamically allocated dataset. To correct this problem, first try resubmitting the job. If the problem persists, you can include your own CWPPRTI DD card in the preprocessor JCL. Be sure to allocate enough primary space to contain your compile listing.

Handling Suppressed Statements

The postprocessor will not support listings that contain suppressed statements. The preprocessor must be run if the listing contains suppressed statements. Suppressed statements include statements that are not generated in the compiler listing due to a %NOPRINT for PL/I. If the postprocessor is run, and it encounters any suppressed statements, you’ll receive the following message, and the postprocessor will stop immediately.

```
PPDRVR130-S POSTPROCESSOR TERMINATED; PREPROCESSOR MUST BE USED DUE TO
%NOPRINT
```

Effect of Optimization Logic on Execution Pointer

While the language processor supports the use of optimized code, it should be noted that occasionally the code execution pointer could follow a different path than expected due to the optimization logic. This consideration only applies to XPEDITER/TSO and XPEDITER/CICS users.

VisualAge PL/I and Enterprise PL/I Enhanced Listings

The PL/I language processor generates an internal line number that is used by XPEDITER/TSO and XPEDITER/CICIS that is different than the “line.file” numbering scheme used by the compiler. If you want to see these line numbers in your listing output, use the PROCESSOR(OUTPUT(PRINT)) command to cause them to be placed to the left of the compiler output.

Usage Restrictions and Examples

This section describes required conditions or conditions not supported by the PL/I language processor and provides examples. Some restrictions apply only to particular products. Other product-specific restrictions may also apply when using the PL/I language processor. Consult your product documentation for more information.

1. Adjustable length variables that are defined in automatic storage are not supported. For example:

   ```
   DCL ARRAY(X,Y) CHAR(1);
   or,
   DCL AREA_VAR AREA(X);
   ```

   In addition to variables with explicitly defined adjustable lengths, any variable with a length that is determined by the compiler to be adjustable is not supported. These variables are denoted in the aggregate length table section in the compiler listing by an 'ADJ' notation in the length columns.

   The IBM compiler flags all entries of an adjustable structure as adjustable, even the elements where the adjustable element starts. Thus, the language processor cannot display these items either.

2. BASED variables with the ALLOCATE statement are not supported. For example:

   ```
   DCL ABCD BIN(31) BASED;
   DCL PTR1 PTR;
   ALLOCATE ABCD SET(PTR1);
   ```

   Because the address value of PTR1 is determined at execution time, the language processor cannot anticipate this value when scanning the compiler listing.

3. Support for multiple unique and separate source programs compiled in a single batch compilation is limited.

   Multiple compiler listings in SYSPRINT (read in as CWPPRTI) are not supported. (There is no batch option for PL/I.)

   Multiple programs that are compiled together in a single job step are not supported. For example:

   ```
   %PROCESS...
   PROC1: PROC OPTIONS(MAIN)
        :
   %PROCESS...
   PROC2: PROC OPTIONS(MAIN)
   ```

4. An array element, or an expression used as a base pointer, is not supported.

   The following examples illustrate these restrictions.

   An array element used as a base pointer:

   ```
   DCL BADDR(5) PTR;
   DCL VARA BASED(BADDR(3));
   ```

   An expression used as a base pointer:

   ```
   DCL VARA BASED(ADDR(BADDR+8));
   ```

   The result of the expression is determined at execution time and cannot be computed by the language processor by scanning the compiler listing.

5. A locator used in a base pointer is not supported. For example:

   ```
   DCL XYZ BASED (ADDR(P1 -> P2));
   or,
   DCL ABC BASED (P3 -> P4);
   ```

The result of the expression is determined at execution time and cannot be computed by the language processor by scanning the compiler listing.

6. AREA variables greater than 32K are not supported. For example:

```
DCL VARA AREA(32768);
```

7. Program names and/or external procedure names that exceed seven characters if the $NIDR parameter is used are not supported. For example:

```
LONG_PROC_NAME:  PROC OPTIONS(MAIN);
DCL LONG_EXTRN_NAME EXTERNAL;
```

   Long procedure names are truncated by the compiler and assigned a new name. The new name assigned will not match the program name of the member in the DDIO file. Compuware recommends that the names chosen should not exceed seven characters.

   For VisualAge PL/I programs only, long procedure names that exceed seven characters are permitted if the CSS shared directory is used.

8. Datanames longer than 30 characters are supported only for PL/I compiler releases 2.3 and later, PL/I ADCYCLE, and PL/I for MVS and VM. Datanames up to 100 characters are supported with VisualAge PL/I and Enterprise PL/I.

   Prior to PL/I compiler release 2.3, the compiler truncates a dataname of 31 characters or longer (which is legal in all compiler releases) to 30 characters when displayed in various sections of the compiler listing (such as the cross-reference section). The language processor cannot determine the true dataname in these situations due to the possibility of duplicates that are only 30 characters in length. The language processor issues a warning message in this situation and processing continues. For compiler releases 2.3 and later, this feature is fully supported.

9. PL/I preprocessor SYSIN and SYSLIB files must be fixed length 80-byte records.

   The compiler allows the SYSIN input stream and included members or macros in SYSLIB to be variable or fixed-length records, up to 100 bytes in length. However, the language processor only supports PL/I programs that use standard 80-byte records.

10. PIC clauses over 163 characters are not supported. For example:

```
DCL BIGPIC PIC 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXX':
```

   If you are using a PIC clause when declaring a variable, the number of characters that make up the clause must not exceed 163.

11. Datanames that are split across two or more lines are not currently supported. For example:

```
DCL READSWITCH CHAR(1) INIT('N');
.
.
IF READSW
  ITCH='Y' THEN DO;
```

   Note: For the VisualAge PL/I compilers and higher, this restriction is removed.

   However, datanames that are qualified can be split at the point of qualification. For example:

```
DCL 1 SWITCHES,
    2 READSWITCH CHAR(1),
    2 WRITESWITCH CHAR(1);

IF SWITCHES.
  READSWITCH='Y' THEN DO;
```

The qualification delimiter character "." must immediately follow the qualifier without intervening spaces.

Note: Usage of the compiler option "Macro" can inhibit the splitting of datanames across lines in the listing.

12. Spaces, when specified within a qualified data name, are not supported.

    For example, the following is invalid:

```
A  .  B =3 ;
```

    No spaces are allowed **between** the first and last variable in the qualification. There is one exception to this restriction; however, CSS can recognize a qualified dataname with spaces when it is split across two lines.

    For example, the following is valid:

```
 A.
B=3;
```

13. Comments in unusual locations, although syntactically correct, will not necessarily be recognized by the language processor. The following is invalid:

```
D/*example comment*/CL ALPHA CHAR(4)
IF ALP/*check alpha name*/HA = 'ABCD' THEN DO;
```

14. Comments started, and not ended on a line that contain a compiler directive when using the postprocessor are not supported. This is not a restriction against multi-line comments. Rather, it is only a restriction when compiler directives and comments are mixed together on the same line, as shown in the following example.

```
%INCLUDE INC1; /*this comment does not end on the same line that
               it started, but instead ends on the second line */
```

15. Labels longer than 31 characters that are defined as elements of an array are not supported. The language processor considers the array dimension as part of the label when determining the total length. For example:

```
DECLARE VERY_LONG_LABEL_ARRAY_NAME(12,6) LABEL;
    :
VERY_LONG_LABEL_ARRAY_NAME(1,1):    /* exactly 31 characters */
    :
VERY_LONG_LABEL_ARRAY_NAME(9,1):    /* exactly 31 characters */
    :
VERY_LONG_LABEL_ARRAY_NAME(12,1):   /* exceeds 31 characters */
```

    Note: For the VisualAge PL/I compilers and higher, this restriction is removed.

16. The NUMBER option is not supported.

    Note: The compiler enables the NUMBER option if GONUMBER or NOSTMT is used.

17. Reserved words used as datanames or labels are not supported. For example:

```
DECLARE DECLARE FIXED BIN(31);
DCL END(2) FIXED;
:
END(1)=DECLARE+1;
   :
```

18. In addition, the following rules apply to %INCLUDE and %NOPRINT statements:

    – The character strings '%INCLUDE' and '%NOPRINT' cannot be split and continued on the next line, for example:

    ```
                                %NOP
    RINT;
    ```

    – %NOPRINT is not supported by the postprocessor.

      Because %NOPRINT prevents sections of code from being written to the compiler listing, the postprocessor cannot scan this code and place it into the DDIO file. The postprocessor stops if it detects this situation since the listing is incomplete. The CSS preprocessor, however, supports the use of %NOPRINT, and processing will proceed normally.

    – If you use PL/I macro processing to generate %INCLUDE or %NOPRINT statements, the macro processing must be performed prior to preprocessing.

      Because the CSS preprocessor scans the source program before compilation, any PL/I macros that internally generate these compiler directives cannot be detected since the macro is not resolved and expanded until compile time. Any undetected compiler directives are then passed to the CSS postprocessor, which stops if they are detected in the compiler listing.

      The best way to resolve this situation, if you must use generated compiler directives, is to preprocess the source program using the PL/I compiler first by using the INCLUDE, MACRO, and NOCOMPILE options. This will generate a source program with all of the macros resolved. This source can then be passed to the CSS preprocessor in a subsequent JCL step for proper scanning and compilation.

19. Source programs or listings with missing commas or syntax errors are not supported. The compiler can detect some of these situations and will make assumptions as to where a misplaced comma may belong. Normally when this happens, the compiler will generate an error message containing "COMMA ASSUMED AFTER ..." or "EXTRANEOUS COMMA DETECTED AFTER ... COMMA IGNORED". The compiler will continue in these cases, but the listing is not changed to reflect the assumption made. Therefore, when the language processor encounters these lines, it may be unable to process correctly.

20. If the postprocessor is used, the listing must be compiled complete with carriage control characters.

Required Options for Supported PL/I Compilers

The following is a list of the PL/I compiler options that must be used with the PL/I language processor.

OS/PLI Compilers:

- AGGREGATE
- ATTRIBUTES(FULL)
- LIST
- MAP
- OFFSET
- OPTIONS
- SOURCE
- STMT
- STORAGE
- SYNTAX
- XREF(FULL)

VisualAge PL/I or Enterprise PL/I Compilers:

- AGGREGATE
- ATTRIBUTES(FULL)
- LIST (required for use with XPEDITER/CICS or XPEDITER/TSO)
- MAP
- NOSTMT
- SOURCE
- SYNTAX
- XREF(FULL)

PL/I Language Processor Command Descriptions and Syntax

This section describes the PL/I language processor commands and their parameters and values.

CONDDDIO (CONDD)

The CONDDDIO command specifies that the source listing will be written to the source listing file unless the compiler return code exceeds **x**. The default is **4**. Permissible values are 0-8.

Note: Unpredictable results may occur when the CONDDDIO parameter is set to **8**.

The following diagram illustrates the syntax for the CONDDDIO command:

```
                  ┌4──┐
►►─CONDDdio(─┼───┼─)──────►◄
                  └x──┘
```

CONDSTEP (CONDS)

The CONDSTEP command controls the condition code set by the postprocessor when it is invoked by the preprocessor. The CONDSTEP parameter is ignored when the postprocessor is invoked as a separate job step after the compile. The postprocessor return code is always used.

The CONDSTEP command can be used only with the preprocessor.

The following diagram illustrates the syntax for the CONDSTEP command:

```
                  ┌HIghest──┐
►►─CONDStep(─┼─────────┼─)──────►◄
                  └COmpiler─┘
```

HIGHEST (HI):

(Default) The condition code will be set to the value of the PL/I language processor return code or the compiler return code, whichever is higher.

COMPILER (CO)

The condition code will always be set to the compiler return code. This option is typically used to allow a subsequent link-edit step to execute when the compile executes normally, but the postprocessor is unable to execute successfully.

DDIO

The DDIO command identifies the information from the PL/I compiler listing that will be written to the DDIO file. The parameter for the DDIO command is OUTPUT. The following diagram illustrates the syntax for the DDIO command OUTPUT parameter.

```
>>-DDio(OUtput(-+-NOCOmpress-+-,-+-XRef-----+-,-+-DMap-----+-,->
                +-COmpress---+   +-NOXRef---+   +-NODMap---+

>-+-LIst-----+-,-+-OFfset-----+-,-+-VARREF-----+-))-><
  +-NOLIst---+   +-NOOFfset---+   +-NOVARREF---+
```

OUTPUT (OU):

Specifies the options for a report file or source listing file that will be written to the DDIO file.

NOCOMPRESS (NOCO):

(Default) Does not compress the compiler listing. The size of the LP source members will not be reduced.

COMPRESS (CO):

Enables compression. Depending on your program size and the options you have set, the size of language processor source members may be reduced.

Note: There are performance implications when compression is used. More CPU and memory is required for compression/decompression of the compiler listing. This is especially important for online usage.

XREF (XR): (Default)

Writes the XREF sections of the listing.

NOXREF (NOXR):

Does not write the XREF sections of the listing.

DMAP (DM):

(Default) Writes the DMAP section of the listing.

NODMAP (NODM):

Does not write the DMAP section of the listing.

LIST (LI):

(Default) Writes the Assembler section of the listing.

NOLIST (NOLI):

Does not write the Assembler section of the listing.

OFFSET (OF):

(Default) Writes the OFFSET section of the listing.

NOOFFSET (NOOF):

Does not write the OFFSET section of the listing.

VARREF:

(Default) Records information in the DDIO file showing where every variable is referenced. This parameter is required and is forced on. Therefore, it cannot be inactivated.

NOVARREF:

This parameter does not cause an error message, but it cannot be set. See VARREF above.

File-Overrides Commands

The File-overrides commands override the default file definitions. The following overrides are available:

- WRK0 - WRK4
- PRTI
- PRTO
- ERRM
- LOAD
- DECK

Note: The **LOAD** or **DECK** ddname is used by the PL/I language processor depending on the options selected for the compile step. If both **LOAD** and **DECK** are specified, both will be processed by the PL/I language processor. If you are using a temporary dataset for your object, and only one option (LOAD or DECK) is specified, eliminate the ddname that corresponds to the option that is NOT specified. For example, if you specify LOAD, then eliminate the DECK ddname.

The File-overrides commands can be used only with the postprocessor.

WRK0 - WRK4 File-Overrides Command

Specifies the work files.

```
►►─WRKn(1)(┌──────────────────────────────────────┐)──►◄
           └─DD(ddname),BLK(blksize),BUF(buffers)─┘
```

Note:

1. In this diagram WRKn represents WRK0 - WRK4.

If you omit DD(ddname), the assumed ddname will be CWPWRKn.

PRTI File-Overrides Command

Specifies the compiler listing input.

```
►►─PRTI(┌──────────────────────────────────────────────────────────────┐)──►◄
        └─DD(ddname),BLK(blksize),BUF(buffers),LRECL(lrecl),RECFM(recfm)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPPRTI.

PRTO File-Overrides Command

Specifies PL/I language processor output.

```
►►─PRTO(─┬──────────────────────────────────────────────────────┬─)──►◄
         └DD(ddname),BLK(blksize),BUF(buffers),LRECL(lrecl)──┘
```

If you omit DD(ddname), the assumed ddname will be CWPPRTO.

ERRM File-Overrides Command

Prints the PL/I language processor error messages.

```
►►─ERRM(─┬──────────────────────────────────────────┬─)──►◄
         └DD(ddname),BLK(blksize),BUF(buffers)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPERRM.

LOAD File-Overrides Command

Specifies the object library (indicated by the LOAD option of the compiler).

```
►►─LOAD(─┬──────────────────────────────────────────┬─)──►◄
         └DD(ddname),BLK(blksize),BUF(buffers)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPLOAD.

DECK File-Overrides Command

Specifies the object library (indicated by the DECK option of the compiler).

```
►►─DECK(─┬──────────────────────────────────────────┬─)──►◄
         └DD(ddname),BLK(blksize),BUF(buffers)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPDECK.

Optional Override Values

The keywords below cause the language processor to override its default values:

DDNAME (DD):

Specify a ddname instead of accepting the default ddname.

BLKSIZE (BLK):

Specify a decimal blocksize instead of accepting the default blocksize.

BUFFERS (BUF):

Specify a decimal number as the number of buffers instead of accepting the default number of buffers, which is **2**. Only the first two buffers are used for buffering input/output. Extra buffers are used as in-core work space for the file. Therefore, records placed in the work space will never be written to the work files. If the storage is available, use this parameter with the BLKSIZE parameter to increase the language processor speed.

If the language processor is running in the MVS/XA or MVS/ESA environment and is above the 16-MB line, a value of zero specifies that as much storage as is available will be used. Using zero places a greater amount of overhead on startup and should not be used for small programs.

LRECL (LR):

Specify a decimal number as the logical record length instead of accepting the default logical record size.

RECFM (RE):

Specify a record format instead of accepting the default record format. Valid formats include:

F	Fixed
FA	Fixed ASA
FM	Fixed Machine
V	Variable
VA	Variable ASA
VM	Variable Machine

INCLUDE (IN)

Use the INCLUDE command to include a library or sequential file member that contains another set of language processor options to be processed.

An included set of options can, in turn, include another set of options. If a recursive chain of INCLUDEs is encountered, the language processor will stop with an error message.

Duplicate or conflicting parameters are resolved as follows:

- Options are processed left to right.
- When an INCLUDE is encountered, the included member is processed immediately. Options following the INCLUDE will be processed after the included member is processed.
- The last option processed takes precedence.
- The sources of options are, in order of descending priority:
  - forced options in the PL/I language compiler installation default module
  - CBL options in the source file (or equivalent for other compilers)
  - PARM options on the EXEC card
  - options specified in the CWPPRMO dataset
  - default options in the PL/I language compiler installation default module.

The INCLUDE command can be used only with the preprocessor.

The following diagram illustrates the syntax for the INCLUDE command:

```
                 ┌CWPPRMO┐
►►─INclude(─┼───────┼─,─┬───────┬─)──►◄
                 └ddname─┘   └member─┘
```

ddname:

The DDNAME of the JCL statement that specifies the dataset name. If a member is not specified, then the ddname must point to a sequential file or library member.

member

The library member name. If the ddname is not specified, then the default CWPPRMO will be used and a comma must precede the member name.

LANGPARM (LANGP)

The LANGPARM command specifies parameters to be passed to the compiler invoked by the language processor. If the language processor is run as a separate job step, this option is ignored.

If you use the CWPPRMO dataset to specify your compiler options, you should use the LANGPARM command. Otherwise, the preprocessor may not be able to distinguish between compiler options and language processor options with the same name. The LANGPARM command allows the preprocessor to identify compiler options that conflict with language processor names, thus preventing unpredictable results.

If you use the PARM statement to specify your compiler options, you can also use the LANGPARM command.

The LANGPARM command can be used only with the preprocessor.

The following diagram illustrates the syntax for the LANGPARM command.

```
>>-LANGParm(-------------------)-------->
            └parm1...parmn-┘
```

parm:

Specifies a parameter to be passed.

LANGUAGE (LANG)

The LANGUAGE command is used to specify the type of compiler for the PL/I language preprocessor and postprocessor.

For the preprocessor, the LANGUAGE command is used to determine which compiler to execute.

For the postprocessor, the LANGUAGE command is used to identify the version of compiler that produced the compiler listing. This is useful when the compiler listing does not show a PL/I header (usually the first line of a listing) that specifies the compiler version.

The following diagram illustrates the syntax for the LANGUAGE command.

```
                 ┌PL/I-----------┐
>>-LANGuage(-----┼-------------┼)-------->
                 ├PLIADcycle-----┤
                 ├VAPL/I---------┤
                 └ENTPL/I--------┘
```

PL/I: (Default)

Specifies that the PL/I compiler should be used.

PLIADCYCLE (PLIAD):

Specifies that the PL/I AD/CYCLE compiler should be used. This is also used for LE/370 PL/I.

VISUALAGEPLI(VAPL/I)

Specifies that the VisualAge PL/I compiler should be used.

ENTERPRISEPLI (ENTPL/I)

Specifies that the Enterprise PL/I compiler should be used.

PL/I

The PL/I command controls the compiler listing destination. The parameter for the PL/I command is OUTPUT. The following diagram illustrates the syntax for the PL/I command OUTPUT parameter.

```
                    ┌NOPRint─┐    ┌NODDio─┐
►►─PL/I(OUtput(─┬─┼────────┼─,─┼───────┼─))──────►◄
                │ └PRint───┘    └DDio───┘ │
                ├BOth─────────────────────┤
                └NONe─────────────────────┘
```

OUTPUT (OU):

Specifies whether the compiler listing will be printed or written to a source listing file. Select from the following values:

NOPRINT (NOPR):

(Default) - Does not print the compiler listing.

PRINT (PR):

Prints the compiler listing.

NODDIO (NODD):

(Default) - Does not write the compiler listing to a source listing file.

DDIO (DD):

Writes the compiler listing to a source listing file.

BOTH (BO):

Prints the listing and writes it to a source listing file. This has the same effect as specifying PL/I(OUTOUT(PRINT,DDIO)).

NONE (NON):

Does not print the compiler listing and does not write it to a DDIO file. This has the same effect as PL/I(OUTPUT(NOPRINT,NODDIO)).

PRINT (PR)

The PRINT command identifies the sections of the listing that will be printed. The parameter for the PRINT command is OUTPUT.

```
                    ┌LIst──┐   ┌OFfset──┐   ┌XRef──┐
►►─PRint(OUtput(─┼──────┼─,─┼────────┼─,─┼──────┼─,──►
                    └NOLIst┘   └NOOFfset┘   └NOXRef┘

   ┌SOurce──┐
►─┼────────┼─))──────►◄
   └NOSOurce┘
```

OUTPUT (OU):

Specifies the options for a source listing that will be printed.

LIST (LI):

(Default) - Prints the Pseudo-Assembler section of the listing.

NOLIST (NOLI):

Does not print the Pseudo-Assembler section of the listing.

OFFSET (OF):

(Default) - Prints the Offset section of the listing.

NOOFFSET (NOOF):

Does not print the Offset section of the listing.

XREF (XR):

(Default) Prints the XREF sections of the listing.

NOXREF (NOXR):

Does not print the XREF sections of the listing.

SOURCE (SO):

(Default) Prints the Source section of the listing.

NOSOURCE (NOSO):

Does not print the Source section of the listing.

PROCESSOR (PROC)

The PROCESSOR command specifies the output parameters of the PL/I language processor. The parameters available for the PROCESSOR command are:

- BYPASS
- ERRORS
- OUTPUT (VISUALAGE PL/I only)
- $NG
- $NIDR
- $NO

BYPASS (BY) Parameter

Specifies whether to run the preprocessor and compiler without running the postprocessor. The following diagram illustrates the syntax for the PROCESSOR command BYPASS parameter.

```
                  ┌NOBYpass─┐
►►─PROCessor(─┼─────────┼─)──►◄
                  └BYpass───┘
```

NOBYPASS (NOBY):

(Default) - Runs the postprocessor.

BYPASS (BY):

Does not run the postprocessor.

ERRORS (ER) Parameter

Specifies how the language processor error messages will be printed. The following diagram illustrates the syntax for the PROCESSOR command ERRORS parameter.

```
                        ┌UPper-case─┐
►►─PROCessor(ERrors(─┼───────────┼─))──►◄
                        └MIxed-case─┘
```

UPPER-CASE (UP):

(Default) - Errors will be printed in uppercase letters.

MIXED-CASE (MI):

Errors will be printed in uppercase and lowercase letters.

OUTPUT (OU) Parameter

Defines options for printing the enhanced listing or writing it to a source listing file. The PLI(OUTPUT(DDIO)) and PROCESSOR(OUTPUT(DDIO)) commands are mutually exclusive. The language processor uses the last-supplied parameter.

Note: This parameter is used by VisualAge PL/I and Enterprise PL/I only.

The following diagram illustrates the syntax for the OUTPUT parameter.

```
                          ┌PRint──┐     ┌NODDio─┐
▶▶─PROCessor(OUtput(─┬─┼───────┼─,─┼───────┼─))──────▶◀
                     │ └NOPrint─┘     └DDio───┘ │
                     ├BOth──────────────────────┤
                     └NONe──────────────────────┘
```

PRINT (PR):

(Default) Prints the enhanced listing.

NOPRINT (NOPR):

Does not print the enhanced listing.

NODDIO (NODD):

(Default) Does not write the enhanced listing to a source listing file.

DDIO (DD):

Writes the enhanced listing to a source listing file.

BOTH (BO):

Has the same effect as specifying PROCESSOR(OUTPUT(PRINT,DDIO)).

NONE (NON):

Does not print the enhanced listing and does not write it to a source listing file.

$NG Parameter

Specifies that the preprocessor's support of INCLUDE %NOPRINT will be bypassed. The following diagram illustrates the syntax for the PROCESSOR command $NG parameter.

```
▶▶─PROCessor(─┬───┬─)──▶◀
              └$NG─┘
```

$NIDR Parameter

Specifies that placement of the date and time stamp on the object module will be bypassed. The following diagram illustrates the syntax for the PROCESSOR command $NIDR parameter.

```
▶▶─PROCessor(─┬─────┬─)──▶◀
              └$NIDR─┘
```

$NO Parameter

Specifies that the preliminary compilation performed by the preprocessor will be bypassed. The preliminary compilation is normally used to determine the compiler options currently in effect. However, if you know that your options satisfy the

requirements of the language processor, you can bypass the preliminary compilation to reduce the preprocessor's execution time.

The following diagram illustrates the syntax for the PROCESSOR command $NO parameter.

```
>>-PROCessor(-+-------+-)-->>
              '-$NO-'
```

PROGRAM-NAME (PRO)

The PROGRAM-NAME command specifies an override name for members in the source listing file. The following diagram illustrates the syntax for the PROGRAM-NAME command.

```
>>-PROgram-name(-+-------+-)-->>
                 '-name-'
```

name:

Specifies an alternative name for a source listing written to the source listing file.

SYSTEM (SYS)

The SYSTEM command performs miscellaneous functions. The syntax for SYSTEM is:

```
>>-SYStem(-+-------+-,-+-------+-,-+-------+-,-+-------+-)-->>
           '-TRace-'   '-TIme-'   '-PTF-'   '-ZAP-'
```

TRACE (TR):

Specifies that internal traces be printed to CWPERRM.

TIME (TI):

Specifies that internal time will be printed with the trace.

PTF:

Specifies that PTF numbers currently installed on the CSS library will be printed. This parameter is used by Compuware Technical Support during problem diagnosis.

ZAP

Specifies that PTF numbers currently installed on the CSS library will be printed. This parameter is used by Compuware Technical Support during problem diagnosis.

Note: The ZAP parameter is allowed for compatibility with pre-SMP/E releases.

Chapter 11.
Assembler Language Processor

A language processor writes a source listing to a VSAM or sequential DDIO file. For Assembler, the source listing includes an assembly listing and other information about a program. This chapter describes the outputs and commands associated with the Assembler language processor.

Operation

The Assembler language processor reads the assembly listing produced by the assembler and writes the listing to the DDIO file and SYSPRINT. The source listing can then be used to provide source support during a debugging or diagnostic session while using any of the following products:

- Abend-AID
- XPEDITER/CICS
- XPEDITER/TSO.

The Assembler language processor can be run as either a preprocessor or postprocessor. Compuware recommends that you use the preprocessor. The postprocessor should only be used in situations where compiled source listings have been stored in machine-readable format.

Preprocessor

Some information about a program module is not always available from the compiler listing. The preprocessor, in addition to gathering information from the compiler listing, gathers information from SYSIN and SYSLIB. The preprocessor provides additional functionality for XPEDITER/CICS and XPEDITER/TSO.

Preprocessor Steps

1. Determines the proper compiler options required to process the listing file.
2. Automatically invokes the compiler or assembler to compile or assemble your source program.
3. Writes the listing to the DDIO file and to SYSPRINT.

Preprocessor Benefits

- **Better handling of compiler errors**. By using the preprocessor, you can avoid the potential problems of processing a listing with compiler errors. The preprocessor, in conjunction with the CONDDDIO parameter, internally checks the return code from the compiler and doesn't write a DDIO listing that contains errors. For example, if you set the CONDDDIO parameter to 8, the preprocessor will write the compiler listing to the DDIO file unless the compiler return code exceeds 8. The preprocessor can process the compiler errors more effectively than the postprocessor.
- **Capturing of suppressed source code**. When the following parameter is used, sections of source code can be suppressed from the compiler listing:

  For Assembler PRINT OFF or PRINT NOGEN

The preprocessor captures this information from SYSIN and SYSLIB.

- **Automated compiler options**. The postprocessor requires that certain compiler options be specified in order to process all needed sections of the compiler listing. The preprocessor can automatically pass the required options to the compiler.

  For a list of the required compiler options, refer "Required Options for Supported Assemblers" on page 11-6.

- **Simplified JCL**. While the postprocessor requires that you add a step after the compile step, the preprocessor requires only that minor modifications be made to your existing compile step.

Dynamically Allocated Files

The following files are dynamically allocated by the Compuware preprocessor. The attributes can be overridden by specifying a DD in the JCL.

Table 11-1. Files Dynamically Allocated by the Preprocessor

DDname	Attributes
CWPWRKn (0-6)	BLKSIZE=19000 SPACE(TRK,(100,80)) UNIT=SYSDA
SORTWK01	BLKSIZE=19000 SPACE(TRK,(100,80)) UNIT=SYSDA
TEMPLIN	BLKSIZE=3200 SPACE(TRK,(200,100)) UNIT=SYSDA
CWPERRM	SYSOUT=*
SYSPRINT	SYSOUT=*
SYSOUT	SYSOUT=*
CWPPRTI	BLKSIZE=16093 SPACE(TRK,(100,200)) UNIT=SYSDA DISP=MOD
CWPPDS2	SPACE (TRK,(100,80,80)) UNIT=SYSDA

CWPPRTO is dynamically allocated as a work file (for the preprocessor). SYSPRINT is where the listing goes.

Postprocessor

The postprocessor is executed as a single step — can be a separate job. It reads in the listing created from the compiler. Information is gathered from the source listing, XREF, data maps, and object code sections of the listing. This information is stored in a DDIO file member and is used by various Compuware products.

While the preprocessor is the preferred method of loading the compiled listing to the DDIO, there are situations that necessitate use of the postprocessor as a viable alternative. This is commonly found in production environments where the listings may be archived. In order for the listings to populate the DDIO, they must be compiled using the required options specified in "Required Options for Supported Assemblers" on page 11-6. Note that these are usually the default options at many shops.

If the programs contain suppressed source code, certain additional control statements must be added to the language processor.

Benefits of Postprocessing

- Exact match of compiled output listings with the executable load module. This is especially important in production environments where time is critical.
- Eliminates the risk of copybook or source code changes prior to recompiling the code since the listing reflects the code that is being executed.
- Significantly less time is needed to process source listings as opposed to recompiling the source code.

Summary of Assembler Language Processor Commands

The following is a list and brief description of the commands available for the Assembler language processor. Each command is discussed in detail later in this chapter.

ASSEMBLER	Controls the assembly listing destination.
CONDDDIO	***(Can only be used with the preprocessor)*** Specifies that the source listing will be written to the source listing file unless the assembly code exceeds the maximum value you set.
CONDSTEP	***(Can only be used with the preprocessor)*** Controls the condition code set by the postprocessor when it is invoked by the preprocessor.
DDIO	Identifies the information from the Assembler assembly listing that will be written to the VSAM or sequential DDIO file.
File-overrides	***(Can only be used with the postprocessor)*** Used to override the default file definitions.
INCLUDE	***(Can only be used with the preprocessor)*** Includes a library or sequential file member that contains another set of language processor options to be processed.
LANGPARM	***(Can only be used with the preprocessor)*** Specifies the parameters to be passed to the assembler invoked by the language processor.
LANGUAGE	Specifies the type of assembler for the Assembler language preprocessor and postprocessor.
PRINT	Identifies the sections of the assembly listing that will be printed.
PROCESSOR	Specifies the output parameters of the Assembler language processor.
PROGRAM-NAME	Specifies an override name for members in the source listing file.
SYSTEM	Performs miscellaneous functions.

The commands can be entered in two ways:

- As parameters on the PARM statement of the EXEC — (Overrides input from the CWPPRMO file.)
- As input from the CWPPRMO file — (Comments are not valid, and all command strings must be within columns 1 through 71.)

Each command has parameters (and some parameters have values) that must be written immediately after the commands, be enclosed in parentheses, and be separated by commas.

If conflicting command options are entered, the last option specified is used.

Usage Considerations

This section includes information that you need to know in order to be able to use the Assembler language processor most efficiently.

Language Processor DD Statements

The following table lists the DD statements for the Assembler language processor.

Table 11-2. Language Processor DD Statements.

DDname	Purpose
CWPWRK (0-6)	LP work files
CWPPRMO	LP parameter input dataset
CWPDDIO	Target source listing DDIO file
CWPPRTI	Compiler listing input to postprocessor
CWPPRTO	Compiler listing output from postprocessor
CWPLOAD	Option LOAD or OBJECT module input to postprocessor
CWPDECK	Option DECK module input to postprocessor
CWPERRM	LP Error Messages

Note: CWPLOAD and CWPDECK correspond to the SYSLIN and SYSPUNCH compiler outputs. If you specify compiler option OBJECT, you should use CWPLOAD in your postprocessor JCL. If you specify compiler option DECK, you should use CWPDECK in your postprocessor JCL. Do not feed the SYSLIN output into CWPDECK or the SYSPUNCH output into CWPLOAD. The updated OBJECT contained in CWPLOAD and CWPDECK should be used in the program's linkedit step. The preprocessor does not use CWPLOAD and CWPDECK.

Assembler Samples and Program Names

The following is a list of members in the CSS installation library that contain sample JCL for executing the Assembler language processor:

Table 11-3. Assembler Sample JCL Members

CXASMPRE	JCL for running the Assembler language preprocessor
CXASM	JCL for running the Assembler language postprocessor
CXLPASM	Sample Assembler language processor options

The following program names are used for the Assembler language processor:

Table 11-4. Assembler Program Names

CWPAMAIN	Assembler preprocessor
CWPADRVR	Assembler postprocessor

Other Usage Considerations

This section describes special considerations for using the Assembler language processor. For a list of Assembler language processor restrictions, refer to "Usage Restrictions" on page 11-6.

Processing Listings with Errors

Unpredictable results may occur when you postprocess an assembly listing with serious errors. If you use the preprocessor, however, you can avoid these potential problems. The preprocessor, in conjunction with the CONDDDIO parameter, internally checks the return code from the assembler and does not write out a listing that contains errors. Using the preprocessor prevents unnecessary processing of an error-filled assembly listing by not calling the postprocessor. For information on the CONDDDIO parameter, refer to "CONDDDIO (CONDD) Command" on page 11-8.

Insufficient Primary Space Allocation

During preprocessing, you may receive a B37 abend on the CWPPRTI DD statement. This is due to insufficient space on the temporary volume used by MVS for the dynamically allocated dataset. To correct this problem, first try resubmitting the job. If the problem persists, you can include your own CWPPRTI DD card in the preprocessor JCL. Be sure to allocate enough primary space to contain your assembly listing.

CAUTION:

If you use a postprocessed source listing with XPEDITER/TSO, you may experience some loss of functionality. If any statement-suppressing directive such as PRINT NOGEN or COPY SUPPRESS is used in postprocessed source, the XPEDITER commands FIND, GEN, KEEP, and PEEK may not function properly.

Handling Suppressed Statements

The postprocessor will not support listings that contain suppressed statements. The preprocessor must be run if the listing contains suppressed statements. Suppressed statements include statements that are not generated in the assembly listing due to a PRINT OFF or a PRINT NOGEN for Assembler. If the postprocessor is run and it encounters any suppressed statements, you will receive the following message, and the postprocessor will stop immediately.

```
PADRVR151-S POSTPROCESSOR TERMINATED; PREPROCESSOR MUST BE USED DUE TO
PRINT OFF/NOGEN
```

You should run the preprocessor if PRINT OFF or PRINT NOGEN is specified. You can bypass this restriction by using the PROCESSOR($ASUP) parameter. This parameter allows a listing with suppressed statements to be postprocessed. You should specify this parameter when you want to postprocess listings with suppressed statements that have been stored in machine-readable format. $ASUP should be used with caution since there may be problems postprocessing a listing containing suppressed statements and using this listing with your Compuware products. For more information on the PROCESSOR($ASUP) parameter, please refer to "PROCESSOR (PROC) Command" on page 11-17.

Usage Restrictions

The following conditions are restrictions on the Assembler language processor.

- The MNOTE instruction statement must begin with an asterisk.
- Assembler preprocessor SYSIN and SYSLIB files must be fixed-length, 80-byte records.
- If more than one dataset is concatenated to SYSIN in the compile step, the first dataset must have the largest blocksize or the DCB=BLKSIZE=XXXXX parameter must be coded.

The following items are NOT supported by the Assembler H or Assembler F language processor:

- Mixed case (LABEL and OPCODE) in source
- The DBCS assembler option
- The NOPRINT parameter specified on the PUSH, POP, or PRINT directive
- The PRINT OFF/NOGEN directive in Assembler-F
- The ICTL directive
- The PRINT OFF/NOGEN directive for the postprocessor.

The following restrictions apply only to the tolerance mode of support for High-Level Assembler. The following items are not supported:

- The NOPRINT parameter specified on the PUSH, POP, or PRINT directive
- PRINT NOMSOURCE
- ALIAS for EXIT, RSECT, START, or DSECT statements
- XOBJECT compiler option
- CEJECT, CATTR, ACONTROL, or AINSERT statements
- GOFF

The following restriction applies when using the postprocessor:

- The listing must be assembled complete with carriage control characters.

Required Options for Supported Assemblers

The following Assembler options must be used with Assembler F, Assembler H, and High Level Assembler 1.1:

- ESD
- LIST

The following Assembler options must be used with High Level Assembler 1.2, 1.3, and 1.4 (for toleration mode support):

- ESD
- LIST - Specify one of the following subparameters. These values change the appearance of the listing.
  - 121 - Creates a listing similar to release 1.
  - 133- Creates a listing with LOC, Addr1, Addr2. MACRO/COPY names on the right side of the listing are expanded to eight characters.
  - MAX - Depending on the LRECL of SYSPRINT, subparameters 121 or 133 will be used.

Assembler LP Command Descriptions and Syntax

This section describes the Assembler language processor commands and their parameters and values.

ASSEMBLER (ASSEM) Command

The ASSEMBLER command controls the assembly listing destination. The parameter for the ASSEMBLER command is OUTPUT.

```
                          ┌NOPRint─┐    ┌NODDio─┐
▶▶─ASSEMbler(OUtput(┬─┬───────┬─,─┬───────┬─┬))──────▶◀
                    │ └PRint──┘    └DDio──┘ │
                    ├BOth──────────────────┤
                    └NONe──────────────────┘
```

OUTPUT (OU):

Specifies whether the assembly listing will print and/or written to a source listing file. Select from the following values:

NOPRINT (NOPR):

(Default) Does not print the assembly listing.

PRINT (PR):

Prints the assembly listing.

NODDIO (NODD):

(Default) Does not write the assembly listing to a source listing file.

DDIO (DD):

Writes the assembly listing to a source listing file.

BOTH (BO):

Prints the listing and writes it to a source listing file. This has the same effect as specifying ASSEMBLER(OUTPUT(PRINT,DDIO)).

NONE (NON):

Does not print the assembly listing and does not write it to a DDIO file. This has the same effect as ASSEMBLER(OUTPUT(NOPRINT,NODDIO)).

CONDDDIO (CONDD) Command

The CONDDDIO command specifies that the source listing will be written to the source listing file unless the assembly return code exceeds **x**. The default is **4**. Permissible values are 0-8.

Note: Unpredictable results may occur when the CONDDDIO parameter is set to 8.

The CONDDDIO command can be used only with the preprocessor.

The following diagram illustrates the syntax for the CONDDDIO command.

```
►►─CONDDdio(─┬─4─┬─)──────►◄
             └─x─┘
```

CONDSTEP (CONDS) Command

The CONDSTEP command controls the condition code set by the postprocessor when it is invoked by the preprocessor. The CONDSTEP parameter is ignored when the postprocessor is invoked as a separate job step after the assembly; the postprocessor return code is always used.

The CONDSTEP command can be used only with the preprocessor.

The following diagram illustrates the syntax for the CONDSTEP command.

```
►►─CONDStep(─┬─HIghest──┬─)──────►◄
             └─COmpiler─┘
```

HIGHEST (HI):

(Default) The condition code will be set to the value of the Assembler language processor return code or the compiler return code, whichever is higher.

COMPILER (CO):

The condition code will always be set to the Assembler return code. This option is typically used to allow a subsequent linkedit step to execute when the assembly executes normally but the postprocessor is unable to execute successfully.

The DDIO command defines the processor DDIO output.

DDIO Command

Identifies the information from the Assembler assembly listing that will be written to the VSAM or sequential DDIO file.

The parameters for the DDIO command are:

- PROGRAM-NAME
- DATE
- TIME
- OUTPUT

PROGRAM-NAME (PRO) Parameter

Specifies that an override name will be used for members in the source listing file. This parameter is not supported by Abend-AID or Abend-AID for CICS.

```
>>─DDio(PROgram-name(┬──────┬))──────────>◄
                     └name──┘
```

name:

Specifies an alternative name for an assembly listing written to the source listing file.

DATE (DAT) Parameter

Overrides the date found in the listing with a different date.

```
>>─DDio(DATe(┬──────┬))──────────>◄
             └date──┘
```

date:

Specifies the alternative date that will be used for the override. The value must be a maximum length of eight characters.

TIME (TI) Parameter

Overrides the time found in the listing with a different time.

```
>>─DDio(TIme(┬──────┬))──────────>◄
             └time──┘
```

time:

Specifies the alternative time that will be used for the override. The value must be a maximum length of eight characters.

OUTPUT (OU) Parameter

Specifies the options for a source listing that is written to the source listing file. The DDIO output parameter defines the sections of the assembler listing that are written to the source listing.

```
                  ┌XRef──┐     ┌NOCOmpress─┐
►►─DDio(OUtput(─┼──────┼─,─┼───────────┼─))──────►◄
                  └NOXRef─┘     └COmpress───┘
```

XREF (XR):

(Default) Writes the XREF sections of the listing.

NOXREF (NOXR):

Does not write the XREF sections of the listing.

NOCOMPRESS (NOCO):

(Default) Does not compress the assembly listing. The size of the language processor members will not be reduced.

COMPRESS (CO):

Enables compression. Depending on the size of your program and the options you have set, the size of language processor source members may be reduced.

Note: There are performance implications when compression is used. More CPU and memory are required for compression/decompression of the assembly listing. This is especially important for online usage.

File-Overrides Commands

The File-overrides commands are used to override the default file definitions. The following overrides are available:

- WRK0 - WRK4
- PRTI
- PRTO
- ERRM
- LOAD
- DECK

Note: The LOAD or DECK ddname is used by the Assembler language processor depending on the options selected for the assembly step. If both LOAD and DECK are specified, both will be processed by the Assembler language processor. If you are using a temporary dataset for your object, and only one option (LOAD or DECK) is specified, eliminate the ddname that corresponds to the option that is NOT specified. For example, if you specify LOAD, then eliminate the DECK ddname.

The File-overrides commands can be used only with the postprocessor.

WRK0 - WRK4 File-Overrides Command

Specifies the work files.

```
►►─WRKn(1)(┌──────────────────────────────────────┐)──►◄
           └─DD(ddname),BLK(blksize),BUF(buffers)─┘
```

Note:

1. In this diagram, WRKn represents WRK0 - WRK4.

If you omit DD(ddname), the assumed ddname will be CWPWRKn.

PRTI File-Overrides Command

Specifies the assembly listing input.

```
►►─PRTI(┌──────────────────────────────────────────────────────────┐)──►◄
        └─DD(ddname),BLK(blksize),BUF(buffers),LRECL(lreci),RECFM(recfm)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPPRTI.

PRTO File-Overrides Command

Specifies Assembler language processor output.

```
►►─PRTO(┌──────────────────────────────────────────────────┐)──►◄
        └─DD(ddname),BLK(blksize),BUF(buffers),LRECL(lreci)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPPRTO.

ERRM File-Overrides Command

Prints the Assembler language processor error messages.

```
►►─ERRM(┬──────────────────────────────────────┬)──►◄
        └DD(ddname),BLK(blksize),BUF(buffers)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPERRM.

LOAD File-Overrides Command

Specifies the object library (indicated by the LOAD option of the assembler).

```
►►─LOAD(┬──────────────────────────────────────┬)──►◄
        └DD(ddname),BLK(blksize),BUF(buffers)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPLOAD.

DECK File-Overrides Command

Specifies the object library (indicated by the DECK option of the assembler).

```
►►─DECK(┬──────────────────────────────────────┬)──►◄
        └DD(ddname),BLK(blksize),BUF(buffers)─┘
```

If you omit DD(ddname), the assumed ddname will be CWPDECK.

Optional Override Values

The keywords below cause the language processor to override its default values:

DDNAME (DD):

Specify a ddname instead of accepting the default ddname.

BLKSIZE (BLK):

Specify a decimal blocksize instead of accepting the default blocksize.

BUFFERS (BUF):

Specify a decimal number as the number of buffers instead of accepting the default number of buffers, which is **2**. Only the first two buffers are used for buffering input/output. Extra buffers are used as in-core work space for the file; therefore, records placed in the work space will never be written to the work files. If the storage is available, use this parameter with the BLKSIZE parameter to increase the language processor speed.

If the language processor is running in the MVS/XA or MVS/ESA environments and is above the 16-MB line, a value of zero specifies that as much storage as is available will be used. Using zero places a greater amount of overhead on startup and should not be used for small programs.

LRECL (LR):

Specify a decimal number as the logical record length instead of accepting the default logical record size.

RECFM (RE):

Specify a record format instead of accepting the default record format. Valid formats include:

F	Fixed
FA	Fixed ASA
FM	Fixed Machine
V	Variable
VA	Variable ASA
VM	Variable Machine

INCLUDE (IN) Command

The INCLUDE command includes a library or sequential file member that contains another set of language processor options to be processed.

An included set of options can, in turn, include another set of options. If a recursive chain of INCLUDEs is encountered, the language processor will stop with an error message.

Duplicate or conflicting parameters are resolved as follows:

- Options are processed left to right.
- When an INCLUDE is encountered, the included member is processed immediately. Options following the INCLUDE will be processed after the included member is processed.
- The last options processed takes precedence.
- The sources of options are, in order of descending priority:
  - forced options in the language assembler installation default module
  - CBL options in the source file (or equivalent for other assemblers)
  - PARM options on the EXEC card
  - options specified in the CWPPRMO dataset
  - default options in the language assembler installation default module

The INCLUDE command can be used only with the preprocessor.

The following diagram illustrates the syntax for the INCLUDE command.

```
                 ┌CWPPRMO┐
►►─INclude(─┼───────┼─,─┬──────┬─)──►◄
                 └ddname─┘   └member─┘
```

ddname:

The ddname of the JCL statement that specifies the dataset name. If a member is not specified, then the ddname must point to a sequential file or library member.

member:

The library member name. If the ddname is not specified, then the default CWPPRMO will be used and a comma must precede the member name.

LANGPARM (LANGP) Command

The LANGPARM command specifies the parameters to be passed to the assembler invoked by the language processor. If the language processor is run as a separate job step, this option is ignored.

If you use the CWPPRMO dataset to specify your assembler options, you should use the LANGPARM command. Otherwise, the preprocessor may not be able to distinguish between assembler options and language processor options with the same name. The LANGPARM command allows the preprocessor to identify compiler options that conflict with language processor names, thus preventing unpredictable results.

If you use the PARM statement to specify your assembler options, you can also use the LANGPARM command.

The LANGPARM command can be used only with the preprocessor.

The following diagram illustrates the syntax for the LANGPARM command.

```
>>-LANGParm(-------------------)-------->
            |                 |
            |parm1...parmn-|
```

parm:

Specifies a parameter to be passed.

LANGUAGE (LANG) Command

The LANGUAGE command specifies the type of assembler for the Assembler language preprocessor and postprocessor.

For the preprocessor, the LANGUAGE command specifies which assembler to execute.

For the postprocessor, the LANGUAGE command identifies the version of assembler that produced the listing. This is useful when the listing does not show an Assembler header (usually the first line of a listing) that specifies the assembler version.

The following diagram illustrates the syntax for the LANGUAGE command.

```
                 ┌HASM──┐
>>-LANGuage(─────┼──────┼─)──────────>
                 ├FASM──┤
                 └HLASM─┘
```

HASM:

(Default) The language processor defaults to an Assembler H listing if the assembler cannot be determined.

FASM:

The language processor assumes an Assembler F listing if the assembler cannot be determined.

HLASM:

The language processor assumes a High Level Assembler listing if the assembler cannot be determined.

PRINT (PR) Command

The PRINT command identifies the sections of the assembly listing that will be printed. The parameter for the PRINT command is OUTPUT.

```
                    ┌XRef──┐    ┌SOurce──┐
►►─PRint(OUtput(─┼──────┼─,─┼────────┼))──────►◄
                    └NOXRef─┘    └NOSOurce─┘
```

OUTPUT (OU):

Specifies the options for a source listing that will be printed.

XREF (XR):

(Default) Prints the XREF sections of the listing.

NOXREF (NOXR):

Does not print the XREF sections of the listing.

SOURCE (SO):

(Default) Prints the Source section of the listing.

NOSOURCE (NOSO):

Does not print the Source section of the listing.

PROCESSOR (PROC) Command

The PROCESSOR command specifies the output parameters of the Assembler language processor. The parameters for the PROCESSOR command are:

- BYPASS
- DIAGNOSE
- ERRORS
- $ASUP
- $NG
- $NIDR
- $NO

BYPASS (BY) Parameter

Specifies whether to run the preprocessor and assembler without running the postprocessor.

```
                  ┌NOBYpass─┐
►►─PROCessor(─┼─────────┼─)──►◄
                  └BYpass───┘
```

NOBYPASS (NOBY):

(Default) Runs the postprocessor.

BYPASS (BY):

Does not run the postprocessor.

DIAGNOSE Parameter

Specifies whether extended diagnostic error messages will be generated.

```
                        ┌NONe──────────────────────────┐
►►─PROCessor(DIAGNOSE(─┼──────────────┬─,─┬──────────────┬─))──►◄
                        │├LOW-core──┤   │├ESA-Index───┤
                        │└NOLOW-core┘   │└NOESA-Index─┘
                        │                                │
                        └ALL─────────────────────────────┘
```

NONE (NON):

(Default) Does not generate either type of message.

LOW-CORE (LOW):

An error message will be generated when an instruction is coded without any base or index register. An error message will be generated for each applicable operand. This option does not perform run time checking of low-core. It only checks object code.

NOLOW-CORE (NOLOW):

The error message will not be generated.

ESA-INDEX (ESA-I):

An error message will be generated when an RX (register-to-index form) instruction is coded with an index register and no base register.

NOESA-INDEX (NOESA-I):

The error message will not be generated.

ALL:

Has the same effect as specifying PROCESSOR(DIAGNOSE(LOW-CORE,ESA-INDEX)).

ERRORS (ER) Parameter

Specifies how the language processor error messages will be printed.

```
                          ┌UPper-case─┐
►►─PROCessor(ERrors(─┼──────────┼─))──────►◄
                          └MIxed-case─┘
```

UPPER-CASE (UP):

(Default) Errors will print in uppercase letters.

MIXED-CASE (MI):

Errors will print in uppercase and lowercase letters.

$ASUP Parameter

Allows a listing with suppressed statements to be postprocessed. For more information on using this parameter, refer to “Handling Suppressed Statements” on page 11-5.

```
►►─PROCessor(─┬──────┬─)──────►◄
              └$ASUP─┘
```

$NG Parameter

Specifies that the preprocessor’s support of PRINT OFF or PRINT NOGEN will be bypassed.

```
►►─PROCessor(─┬──────┬─)──────►◄
              └$NG───┘
```

$NIDR Parameter

Specifies that placement of the date and time stamp on the object module will be bypassed.

```
►►─PROCessor(┬──────┬─)──►◄
             └$NIDR─┘
```

$NO Parameter

Specifies that the preliminary assembly performed by the preprocessor will be bypassed. The preliminary assembly is normally used to determine the assembler options currently in effect. However, if you know that your options satisfy the requirements of the language processor, you can bypass the preliminary assembly to reduce the preprocessor's execution time.

```
►►─PROCessor(┬─────┬─)──►◄
             └$NO─┘
```

PROGRAM-NAME (PRO) Command

The PROGRAM-NAME command specifies an override name for members in the source listing file. The following diagram illustrates the syntax for the PROGRAM-NAME command.

```
►►─PROgram-name(┬──────┬─)──►◄
                └name─┘
```

name:

Specifies an alternative name for a source listing written to the source listing file.

SYSTEM (SYS) Command

The SYSTEM command performs miscellaneous functions. The following diagram illustrates the syntax for the SYSTEM command.

```
>>-SYStem(┬────┬,┬────┬,┬────┬,┬────┬)-><
          └TRace┘ └TIme┘ └PTF┘ └ZAP┘
```

TRACE (TR):

Specifies that the internal trace will print to CWPERRM.

TIME (TI):

Specifies that the internal time will print with the trace.

PTF:

Specifies that the PTF numbers currently installed on the CSS library will be printed. This parameter is used by Compuware Technical Support during problem diagnosis.

ZAP:

Specifies that the PTF numbers currently installed on the CSS library will be printed. This parameter is used by Compuware Technical Support during problem diagnosis.

Note: The ZAP parameter is allowed for compatibility with pre-SMP/E releases.

Chapter 12.
C Language Processor

A language processor writes a source listing to a VSAM or sequential DDIO file. The DDIO file may be one of several DDIO files attached to a shared directory. For C, the source listing includes the compiler listing and other information about the program.

In addition, a language processor updates the object file (SYSLIN) produced by the compiler. The information added by the language processor enables XPEDITER/TSO to locate the correct source listing file in DDIO when you later debug the compiled program.

This chapter describes the outputs and commands associated with the C language processor.

Operation

The C language processor reads the compiler listing produced by the C compiler and writes the compiler listing to the DDIO file and SYSCPRT. The source listing in DDIO can then be used to provide source support during a debugging or diagnostic session using XPEDITER/TSO.

The C language processor can be run as either a preprocessor or postprocessor. The preprocessor sets up the environment for the C compiler, then invokes the C compiler and captures and analyzes the compiler listing. The postprocessor captures and analyzes the listing from a previous run of the C compiler. Compuware recommends that you use the preprocessor. The postprocessor should only be used in situations where compiled source listings have been stored in machine-readable format.

Preprocessor

Preprocessor Steps

1. Determines the proper compiler options required to process the listing file.
2. Automatically invokes the compiler or assembler to compile or assemble your source program.
3. Writes the listing to the DDIO file and to SYSCPRT.

Preprocessor Benefits

- **Better handling of compiler errors**. By using the preprocessor, you can avoid the potential problems of processing a listing with compiler errors. The preprocessor, in conjunction with the CONDDDIO parameter, internally checks the return code from the compiler and doesn't write a DDIO listing that contains errors. For example, if you set the CONDDDIO parameter to 4, the preprocessor will write the compiler listing to the DDIO file unless the compiler return code exceeds 4. The preprocessor can process the compiler errors more effectively than the postprocessor.
- **Capturing of suppressed source code.** When the following parameter is used, sections of source code can be suppressed from the compiler listing:

  For C NOSHOWINC or NOEXPMAC

  The preprocessor captures this information by forcing on more revealing compiler options.

- **Automated compiler options**. The postprocessor requires that certain compiler options be specified in order to process all needed sections of the compiler listing. The preprocessor can automatically pass the required options to the compiler.
- **Simplified JCL**. While the postprocessor requires that the user add a step after the compile step, the preprocessor requires only that minor modifications be made to your existing compile step.

Dynamically Allocated Files

The following files are dynamically allocated by the Compuware preprocessor. The attributes can be overridden by specifying a DD in the JCL.

Table 12-1. Files Dynamically Allocated by the Preprocessor

DDname	Attributes
CWPWRKn (0-6)	BLKSIZE=19000 SPACE(TRK,(100,80)) UNIT=SYSDA
TEMPLIN	BLKSIZE=3200 SPACE(TRK,(200,100)) UNIT=SYSDA
CWPERRM	SYSOUT=*
SYSPRINT	SYSOUT=*
SYSOUT	SYSOUT=*
CWPPRTI	BLKSIZE=16093 SPACE(TRK,(100,200)) UNIT=SYSDA DISP=MOD

CWPPRTO is dynamically allocated as a work file (for the preprocessor). SYSPRINT is where the listing goes.

Postprocessor

The postprocessor is executed as a single step — can be a separate job. It reads in the listing created from the compiler. Information is gathered from the source listing, XREF, data maps, and object code sections of the listing. This information is stored in a DDIO file member and is used by various Compuware products.

While the preprocessor is the preferred method of loading the compiled listing to the DDIO, there are situations that necessitate use of the postprocessor as a viable alternative. This is commonly found in production environments where the listings may be archived. In order for the listings to populate the DDIO, they must be compiled using the required options specified in “Required Options for Supported C Compilers” on page 12-7. Note that these are usually the default options at many shops.

Benefits of Postprocessing

- Exact match of compiled output listings with the executable load module. This is especially important in production environments where time is critical.
- Eliminates the risk of copybook or source code changes prior to recompiling the code since the listing reflects the code that is being executed.
- Significantly less time is needed to process source listings as opposed to recompiling the source code.

Summary of C Language Processor Commands

The following is a list and brief description of the commands available for the C language processor. Each command is discussed in detail later in this chapter.

C	Controls whether the language processor should print the compiler listing and/or write it to DDIO.
CONDDDIO	***(Can only be used with the preprocessor)*** Specifies that the source listing will be written to the DDIO file unless the compiler return code exceeds the maximum value you set.
CONDSTEP	***(Can only be used with the preprocessor)*** Controls the condition code set by the preprocessor.
DDIO	Specifies which sections of the C compiler listing will be written to the DDIO file.
File-overrides	***(Can only be used with the postprocessor)*** Override certain default file definitions (ddnames) used by the language processors.
INCLUDE	Includes a library or sequential file member that contains another set of language processor options to be processed.
LANGPARM	***(Can only be used with the preprocessor)*** Specifies parameters to be passed to the compiler invoked by the language processor.
LANGUAGE	Specifies the type of compiler for the C language preprocessor and postprocessor.
PRINT	Identifies the sections of the listing that will be printed.
PROCESSOR	Specifies the output parameters of the C language processor.
PROGRAM-NAME	Specifies an override name for members in the source listing file.
SYSTEM	Performs miscellaneous functions.

The commands can be entered in two ways:

- As parameters on the PARM statement of the EXEC — (Override input from the CWPPRMO file.)
- As input from the CWPPRMO file — (Comments are not valid, and all command strings must be within columns 1 through 71.)

Each command has parameters (and some parameters have values) that must be written immediately after the commands, be enclosed in parentheses, and separated by commas.

If conflicting command options are entered, the last option specified is used.

Usage Considerations

This section includes information that you need to know in order to be able to use the C language processor most efficiently.

DDIO Member Name

The C language processors write the listing as a member of a DDIO file. The processors select a member name by applying the following tests (in the order shown).

1. If the PROGRAM-NAME(name) option was specified in CWPPRMO or on the EXEC PARM= parameters, that name is used as the DDIO member name.
2. If your program includes a '#pragma csect(CODE,"name")' directive, the name specified for the CSECT is used as the DDIO member name.
3. If your source program was read from a member of a PDS or PDSE, that member name is used as the DDIO member name.
4. If your program includes any external-linkage (non-static) functions, the name of the first such function is used as the DDIO member name.
5. If your program includes any external-linkage (file scope, non-static) variables, the name of the first such variable is used as the DDIO member name.
6. Failing all of the above, "TEMPNAME" will be used as the DDIO member name.

If the DDIO member name thus selected is longer than eight (8) characters, you must write the listing to a Source Listing Shared Directory rather than an ordinary DDIO file. See **Chapter 7, "Batch File Utility CWDDLPUT"** for a description of how to create a Source Listing Shared Directory.

Language Processor DD Statements

The following table lists the DD statements for the C language processor:

Table 12-2. Language Processor DD Statements.

DDname	Purpose
CWPPRMO	LP parameter input dataset
CWPDDIO	Target source listing DDIO file
CWPPRTI	Compiler listing input to postprocessor
CWPPRTO	Compiler listing output from postprocessor
CWPLOAD	OBJECT module input to postprocessor
CWPERRM	LP Error Messages

C Samples and Program Names

The following is a list of members in the CSS installation library that contain sample JCL for executing the C language processor:

Table 12-3. C Sample JCL Members

CXCPRE	JCL for running the C language preprocessor
CXC	JCL for running the C language postprocessor
CXC2	JCL for running the C language postprocessor with compiler listings stored in machine-readable format
CXLPC	Sample C language processor options

The program names of the two C language processors are:

Table 12-4. C Program Names

CWPZMAIN	C preprocessor
CWPZDRVR	C postprocessor

Other Usage Considerations

This section describes special considerations for using the C language processor.

Processing Listings with Errors

Unpredictable results may occur when you postprocess a compiler listing with serious errors. However, if you use the preprocessor, you can avoid these potential problems. The preprocessor, in conjunction with the CONDDDIO parameter, internally checks the return code from the compiler and does not write out a listing that contains errors. Using the preprocessor prevents unnecessary processing of an error-filled compiler listing by not calling the postprocessor if the compiler error level is too high.

Insufficient Primary Space Allocation

During preprocessing, you may receive a B37 abend on the CWPPRTI DD statement. This is due to insufficient space on the temporary volume used by MVS for the dynamically allocated dataset. To correct this problem, first try resubmitting the job. If the problem persists, you can include your own CWPPRTI DD card in the preprocessor JCL. Be sure to allocate enough primary space to contain your compile listing.

Effect of Optimization Logic on Execution Pointer

While the language processor supports the use of optimized code, it should be noted that occasionally the code execution pointer could follow a different path than expected due to the optimization logic. This consideration only applies to XPEDITER users.

Batch Compile of Multiple C Programs in a PDS

The Compuware C preprocessor does not support allocating an entire PDS to SYSIN DD, with the result that every member of the PDS is compiled. You can compile only one input file (sequential dataset or PDS member) in one run of the Compuware preprocessor.

Use of HFS Files

The Compuware C language processors do not support the use of HFS (the Unix file system) for any input or output files.

Use of the OPTFILE Compiler Option to Specify Any Option Required by the Language Processors

The Compuware preprocessor supports the use of the OPTFILE compiler option, which allows you to include compiler options from specified files, but the preprocessor does not examine the contents of these files as they are included. If, for example, you specified the SOURCE option only in an OPTFILE, the Compuware preprocessor would presume that you were using the default NOSOURCE option, and would alter the listing sent to SYSCPRT to emulate NOSOURCE. If you want to specify compiler options or language processor options in external files, consider including these files with the INCLUDE option in CWPPRMO input to the language processors, since the language processors do examine the contents of INCLUDE files.

Usage Restrictions

This section describes conditions not yet supported by the C language processor.

Other product-specific restrictions may also apply when using the C language processor. Consult your CSS-related product documentation for more information.

The following conditions are not currently supported:

1. Inconsistent use of the #line directive.

   The #line directive of the C/C++ compiler allows the programmer to assert that subsequent lines of source will be considered as included from a named external file (which need not exist). The Compuware C language processors support the use of #line, as long as it does not result in two or more different lines of source being attributed to the same line number of the same external file, or to the same line number of the primary input file.

2. Use of pre-compiled headers.

   The Compuware preprocessor will automatically append the options NOGENPCH and NOUSEPCH to the options passed to the compiler.

3. Use of #pragma to change compiler options.

   Use of “#pragma options” to change compiler options dynamically is not supported for certain options. Specifically, you must not specify any of the following in a “#pragma options(...)” directive:

   - XREF
   - NOXREF
   - OBJECT
   - NOOBJECT

4. Use of the OPTFILE compiler option to specify any option required by the language processors.

   The Compuware preprocessor supports the use of the OPTFILE compiler option, which allows you to include compiler options from specified files, but the preprocessor does not examine the contents of these files as they are included. If, for example, you specified the SOURCE option only in an OPTFILE, the Compuware preprocessor would presume that you were using the default NOSOURCE option and would alter the listing sent to SYSCPRT to emulate NOSOURCE. To specify compiler options or language processor options in external files, consider including these files with the INCLUDE option in CWPPRMO input to the language processors since the language processors do examine the contents of INCLUDE files.

5. Redirection of the compiler listing by specification of an alternate dataset or ddname on either the LISTING or INLRPT option is not supported.

6. The Compuware C language processor does not support all the language constructs permitted by the IBM compiler when using compiler option LANGLVL(COMMONC). For example, anonymous unions are not supported.

The following restriction applies when using the postprocessor:

- If the postprocessor is used, the listing must be compiled complete with carriage control characters.

Required Options for Supported C Compilers

The following is a list of the C compiler options that must be used with the C language processor:

- EXPMAC
- LIST
- SHOWINC
- SOURCE
- XREF

If you use the Compuware preprocessor to compile your C programs, it will automatically invoke the compiler with these options, then adjust the listing to match the options specified in your JCL. If you use the Compuware postprocessor to process a listing that was not compiled with all these options, you will receive the message:

PZDRVR305 – S Compiler option OPTION is required

where OPTION is the required option that is missing. The postprocessor will stop with an error return code.

C Language Processor Command Descriptions & Syntax

This section describes the C language processor commands, along with their parameters and values.

C

The C command controls whether the language processor should print the compiler listing and/or write it to DDIO.

The following diagram illustrates the syntax for the C command. You must specify OUTPUT as the parameter of the C command.

```
                 ┌NOPRint─┐    ┌NODDio─┐
►►─C(OUtput(─┬─┼────────┼─,─┼───────┼─┬))──►◄
             │ └PRint───┘    └DDio───┘ │
             ├BOth──────────────────────┤
             └NONe──────────────────────┘
```

OUTPUT (OU):

Specifies whether the compiler listing will be printed or written to a source listing DDIO file. Select from the following values:

NOPRINT (NOPR):

(Default) Does not print the compiler listing.

PRINT (PR):

Prints the compiler listing.

NODDIO (NODD):

(Default) Does not write the compiler listing to a source listing file.

DDIO (DD):

Writes the compiler listing to a source listing file.

BOTH (BO):

Prints the listing and writes it to a source listing file. This has the same effect as specifying C(OUTPUT(PRINT,DDIO)).

NONE (NON):

Does not print the compiler listing and does not write it to a DDIO file. This has the same effect as C(OUTPUT(NOPRINT,NODDIO)).

CONDDDIO (CONDD)

The CONDDDIO command specifies that the source listing will be written to the DDIO file unless the compiler return code exceeds **x**. The default is **4**. Permissible values are 0-4.

The CONDDDIO command can be used only with the preprocessor.

The following diagram illustrates the syntax for the CONDDDIO command:

```
                  ┌4──┐
►►─CONDDdio(─────┼───┼─)──────►◄
                  └x──┘
```

CONDSTEP (CONDS)

The CONDSTEP command controls the condition code set by the preprocessor. The CONDSTEP parameter is ignored when the postprocessor is invoked as a separate job or job step.

The CONDSTEP command can be used only with the preprocessor.

The following diagram illustrates the syntax for the CONDSTEP command:

```
                  ┌HIghest──┐
►►─CONDStep(─────┼─────────┼─)──────►◄
                  └COmpiler─┘
```

HIGHEST (HI):

(Default) The condition code will be set to the value of the C language processor return code or the compiler return code, whichever is higher.

COMPILER (CO):

The step condition code will always be set to the compiler return code. This option is typically used to allow a subsequent linkedit step to execute when the compile executes normally, but the language processor's capture or analysis of the listing is unable to execute successfully.

DDIO

The DDIO command specifies which sections of the C compiler listing will be written to the DDIO file.

Specify the sections of the compiler listing using the OUTPUT parameter of the DDIO command. The following diagram illustrates the syntax for the DDIO command.

```
                  ┌XRef──┐    ┌LIst──┐
►►─DDIO(OUtput(─┼──────┼─,─┼──────┼─))──────►◄
                  └NOXRef─┘    └NOLIst─┘
```

OUTPUT (OU):

Specifies which sections of the compiler listing will be written to the DDIO file.

XREF (XR):

(Default) Writes the XREF sections of the listing.

NOXREF (NOXR):

Does not write the XREF sections of the listing.

LIST (LI):

(Default) Writes the Pseudo-Assembler section of the listing.

NOLIST (NOLI):

Does not write the Pseudo-Assembler section of the listing.

File-Overrides Commands

The File-overrides commands override certain default file definitions (ddnames) used by the language processors. These ddnames are used by both the preprocessor and the postprocessor:

- **CWPERRM**
- **CWPPRTI**
- **CWPPRTO**
- **CWPLOAD**

All of these ddnames except CWPERRM must be specified when you run the postprocessor. None of them is required when you run the preprocessor. Any ddname not specified by you is allocated dynamically by the language processor.

Any of these four ddnames can be overridden by a file override command. The File-overrides commands can be used only with the postprocessor.

The following overrides are available:

ERRM File-Overrides Command

Error and informational messages from the language processor.

The following diagram illustrates the syntax for the ERRM file-override command:

```
>>-ERRM(-+-------------------------------+-><
         +-DD(ddname),BLK(blksize)-+
```

If you omit DD(ddname), the assumed ddname will be CWPERRM.

PRTI File-Overrides Command

The compiler listing to be analyzed.

The following diagram illustrates the syntax for the PRTI file-override command:

```
>>-PRTI(-+-------------------------------------------------------+-)-><
         +-DD(ddname),BLK(blksize),LRECL(lrecl),RECFM(recfm)-+
```

If you omit DD(ddname), the assumed ddname will be CWPPRTI.

PRTO File-Overrides Command

The compiler listing printed after the language processor has completed.

The following diagram illustrates the syntax for the PRTO file-override command:

```
>>-PRTO(-+-------------------------------+-)-><
         +-DD(ddname),BLK(blksize)-+
```

If you omit DD(ddname), the assumed ddname will be CWPPRTO.

LOAD File-Overrides Command

The object file created by the compiler, which the language processor will modify.

The following diagram illustrates the syntax for the LOAD file-override command:

```
►►─LOAD(─┬──────────────────────────┬─)──────►◄
         └─DD(ddname),BLK(blksize )─┘
```

If you omit DD(ddname), the assumed ddname will be CWPLOAD.

Optional Override Values

The keywords below cause the language processor to override its default values.

DDNAME (DD):

Specify a ddname instead of accepting the default ddname.

BLKSIZE (BLK):

Specify a decimal blocksize instead of accepting the default blocksize.

LRECL (LR):

Specify a decimal number as the logical record length instead of accepting the default logical record size.

RECFM (RE):

Specify a record format instead of accepting the default record format. Valid formats include:

F	Fixed
FA	Fixed ASA
FM	Fixed Machine
V	Variable
VA	Variable ASA
VM	Variable Machine

INCLUDE (IN)

The INCLUDE command includes a library or sequential file member that contains another set of language processor options to be processed.

An included set of options can, in turn, include another set of options. If a recursive chain of INCLUDEs is encountered, the language processor will ignore the INCLUDE that caused the recursion, and continue processing.

Duplicate or conflicting parameters are resolved as follows:

- Options are processed left to right.
- When an INCLUDE is encountered, the included member is processed immediately. Options following the INCLUDE will be processed after the included member is processed.
- The last option processed takes precedence.
- The sources of options are, in order of descending priority:
  - forced options in the C language compiler installation default module
  - #pragma option() options in the source file
  - PARM options on the EXEC card
  - options specified in the CWPPRMO dataset
  - default options in the C language compiler installation default module

The following diagram illustrates the syntax for the INCLUDE command:

```
                 ┌CWPPRMO┐
►►─INclude(─┼───────┼─,─┬───────┬─)──►◄
                 └ddname─┘   └member─┘
```

ddname:

The DDNAME of the JCL statement that specifies the dataset name. If a member is not specified, then the ddname must point to a sequential file or library member.

member:

The library member name. If the ddname is not specified, then the default CWPPRMO will be used and a comma must precede the member name.

LANGPARM (LANGP)

The LANGPARM command specifies parameters to be passed to the compiler invoked by the language processor. The postprocessor ignores this option.

If you use the CWPPRMO dataset to specify your compiler options, you should use the LANGPARM command. Otherwise, the preprocessor may not be able to distinguish between compiler options and language processor options with the same name. The LANGPARM command allows the preprocessor to identify compiler options that conflict with language processor names, thus preventing unpredictable results.

If you use the PARM statement to specify your compiler options, you can also use the LANGPARM command.

The LANGPARM command is meaningful only when used with the preprocessor.

The following diagram illustrates the syntax for the LANGPARM command:

```
>>-LANGParm(-------------------)-------><
            └parm1...parmn┘
```

parm:

Specifies a parameter to be passed.

LANGUAGE (LANG)

The LANGUAGE command specifies the type of compiler for the C language preprocessor and postprocessor.

For the preprocessor, the LANGUAGE command determines which compiler to execute.

For the postprocessor, the LANGUAGE command identifies the version of compiler that produced the compiler listing. This is useful when the compiler listing does not show a header (usually the first line of a listing) that specifies the compiler version.

The following diagram illustrates the syntax for the LANGUAGE command:

```
                 ┌C390──┐
>>-LANGuage(-┼──────┼-)-------------><
                 ├CMVS──┤
                 └CZOS──┘
```

C390:

(Default) Specifies that some version of the C/C++ for OS/390 compiler should be used. The preprocessor will invoke program CBCDRVR, which is an alias for the compiler module in the libraries distributed for every version of C/C++ for OS/390. You should include the proper libraries for the compiler version you want to use in the STEPLIB DD of your JCL.

CMVS:

Specifies that the C/C++ for MVS compiler (CBC320PP) should be used.

CZOS

Specifies that the z/OS 1.2 or higher C/C++ compiler (CCNDRVR) should be used.

PRINT (PR)

The PRINT command identifies the sections of the listing that will be printed.

The following diagram illustrates the syntax for the PRINT command. You must specify OUTPUT as the parameter of the PRINT command.

```
                  ┌XRef──┐   ┌LIst───┐   ┌SOurce──┐
►►─PRint(OUtput(─┼──────┼─,─┼───────┼─,─┼────────┼─))──►◄
                  └NOXRef┘   └NOLIst──┘   └NOSOurce┘
```

OUTPUT (OU):

Specifies the options for a source listing that will be printed.

XREF (XR):

(Default) Prints the XREF sections of the listing.

NOXREF (NOXR):

Does not print the XREF sections of the listing.

LIST (LI):

(Default) Prints the Pseudo-Assembler section of the listing.

NOLIST (NOLI):

Does not print the Pseudo-Assembler section of the listing.

SOURCE (SO):

(Default) Prints the Source section of the listing.

NOSOURCE (NOSO):

Does not print the Source section of the listing.

PROCESSOR (PROC)

The PROCESSOR command specifies the output parameters of the C language processor. The parameters available for the PROCESSOR command are:

ERRORS (ER) Parameter

Specifies how the language processor error messages will be printed.

The following diagram illustrates the syntax for the PROCESSOR command ERRORS parameter.

```
                          ┌UPper-case─┐
►►─PROCessor(ERrors(─┼──────────┼─))──────►◄
                          └MIxed-case─┘
```

UPPER-CASE (UP):

(Default) Errors will be printed in uppercase letters.

MIXED-CASE (MI):

Errors will be printed in uppercase and lowercase letters.

$NIDR Parameter

Specifies that placement of the date and time stamp on the object module will be bypassed.

The following diagram illustrates the syntax for the PROCESSOR command $NIDR parameter.

```
►►─PROCessor(─┬──────┬─)──────►◄
              └$NIDR─┘
```

$NO Parameter

Specifies that the preliminary compilation performed by the preprocessor will be bypassed. The preliminary compilation normally determines the compiler options currently in effect. However, if you know that your options satisfy the requirements of the language processor, you can bypass the preliminary compilation to reduce the preprocessor's execution time.

The following diagram illustrates the syntax for the PROCESSOR command $NO parameter.

```
►►─PROCessor(─┬──────┬─)──────►◄
              └$NO───┘
```

PROGRAM-NAME (PRO)

The PROGRAM-NAME command specifies an override name for members in the source listing file.

The following diagram illustrates the syntax for the PROGRAM-NAME command:

```
>>-PROgram-name(-------------)-------><
                 └name┘
```

name:

Specifies an alternative name for a source listing written to the source listing file.

SYSTEM (SYS)

The SYSTEM command performs miscellaneous functions.

The following diagram illustrates the syntax for the SYSTEM command:

```
>>-SYStem(-------,-------)-><
          └PTF┘   └ZAP┘
```

PTF:

Specifies that the PTF numbers currently installed on the CSS library will be printed. This parameter is used by Compuware Technical Support during problem diagnosis.

ZAP:

Specifies that the PTF numbers currently installed on the CSS library will be printed. This parameter is used by Compuware Technical Support during problem diagnosis.

Note: The ZAP parameter is allowed for compatibility with pre-SMP/E releases.

Chapter 13.
CSS Security Exit

The CSS Security Exit program is a user-coded program for Compuware products that use any of the following CSS components:

- CWDDSUTL batch file utility
- CWFXSDUT batch file utility (certain commands only)
- CWAASDUT batch file utility (certain commands only)
- CWDDLPUT batch file utility (certain commands only)
- Abend-AID for CICS online viewing facility
- AADF (a CICS online viewing facility used by Abend-AID for CICS)

This chapter describes how to prepare the exit.

Note: The name of the Security Exit program must be CWASSECU. A sample can be found in the CSS SLCXCNTL installation library, member CWASSECU.

Operation

The Security Exit program enables you to control access to CSS DDIO file members by controlling access to the user command. Using a security exit program is optional. The Compuware Security Exit program can be used in conjunction with your existing security package to secure sensitive data used by the following Compuware products:

- Abend-AID (VisualAge PL/I and Enterprise PL/I language support is also available)
- Abend-AID Fault Manager
- Abend-AID for WebSphere MQ
- Abend-AID for CICS
- STROBE
- XPEDITER/CICS (VisualAge PL/I and Enterprise PL/I language support is also available)
- XPEDITER/Code Coverage
- XPEDITER/TSO and XPEDITER/IMS (VisualAge PL/I and Enterprise PL/I language support is also available)

After the Security Exit program has been coded for any one of these Compuware products, it will also be called by the remaining listed products if they are installed at your site and share the same load library.

If the Security Exit program is present, it is called for each member selected. The Security Exit program determines if the requested command is permitted for the user, and then it returns the appropriate code to the calling facility (CWDDSUTL, CWFXSDUT, CWAASDUT, CWDDLPUT, Abend-AID for CICS server, or AADF). If the command is unacceptable for one of the selected members, it will not be executed at all.

Security Exit Program Functions

The functioning of the Security Exit program depends on the type of DDIO file. The exit, as distributed, has the following default permissions:

Abend-AID report files:

- Permits only the owner of a report file to delete, view, or print the report via batch processing.
- Prohibits anyone from formatting production datasets.
- All other commands are available to all users.

Abend-AID for CICS transaction databases:

- Allows general viewing access to the report file or transaction database directory.
- Allows viewing of a report file or transaction database by an operator at abend time or by a programming manager at any time.
- Prohibits online deletion of diagnostic reports or transaction reports.
- Permits all commands except DELETE for batch processing of report files or transaction databases.

Source Listing Files:

- All commands are available to users.

Abend-AID for CICS Security Exit Considerations

The Security Exit program allows you to restrict access to some of the CWFXSDUT functions for transaction databases and source listing files. For transaction databases, you can use the Security Exit program to restrict access to individual dumps within a transaction database. When the Compuware Security Exit program and/or an external security package is used, security is enabled as described in the following table:

Table 13-1. Security Exit Enablement for Abend-AID for CICS Commands

	Transaction Databases	Source Listing Files
Online	All commands	All commands except DIR
Batch	CWFXSDUT commands COPY, DELETE, EXPORT, IMPORT, LOCK, MOVE, UNLOCK	All CWDDSUTL commands

Note: As already mentioned, after the Compuware Security Exit program is enabled for any of the listed Compuware products except Abend-AID for CICS, it is automatically enabled for the other listed Compuware products.

After your external security package has verified that the server region and the user can access the dataset, security checking is performed as follows:

- **Transaction database**

  If the Compuware Security Exit program is present and the Abend-AID for CICS EXTERNAL_SECURITY_DDIO_METHOD server configuration parameter is CSS, the Compuware Security Exit is called and any external security package is ignored. If the Compuware Security Exit program is not present, or the server configuration parameter EXTERNAL_SECURITY_DDIO_METHOD equals SAF, the external security package is called.

- **Source listing file**

  When you are using the Abend-AID for CICS online viewing facility (through ISPF, AADF, or VTAM) and an attempt is made to open a source listing file, the Compuware Security Exit program is called to determine whether the user is authorized for the requested command. (This is only true if the Abend-AID for CICS server configuration parameter EXTERNAL_SECURITY_DDIO_METHOD equals DDIO.)

  When an attempt is made to modify the source listing file, the Compuware Security Exit program is called to determine whether the user is authorized to use the requested command.

Note: When using the Compuware Security Exit program, if the Security Exit program module is replaced while a Abend-AID for CICS server region is running, the new copy will not be available until the region is recycled.

Security Exit Program Inputs and Outputs

Upon entering the Security Exit program, the registers must be saved. A save area is provided as part of the command list pointed to by R1 (field SECUAREA). The registers must be restored prior to returning to the original calling facility (CWDDSUTL, CWFXSDUT, CWDDLPUT, Abend-AID for CICS server, or AADF). An exception is R15, which holds the return code or address of the customer-supplied message that provides the calling facility with the security check result.

Note: Any services native to the environment, such as CICS command-level services, cannot be developed within this exit.

Input Information

The contents of the registers upon entry to the Security Exit program are:

R1 Contains the command list address.

R13 Contains the Save Area address passed by the caller.

R14 Contains the return address to the calling facility.

R15 Contains the entry point address for the CWASSECU Security Exit program.

Command List

The command list passed to the Security Exit program is mapped by CWASSECD DSECT, which is defined by the CWASSECD macro in SLCXCNTL installation library member CWASSECD, and is shown in Figure 13-1 on page 13-5.

Each field in the CWASSECD DSECT is followed by a letter code describing the type of file for which the field is filled. The codes are:

C Abend-AID for CICS transaction database

E Source listing file (Abend-AID, Abend-AID for CICS, XPEDITER/CICS, XPEDITER/ TSO)

O Abend-AID report file

A All of the above

The SECUCUSR field contains the USERID when invoked from a batch job using CWDDSUTL or CWFXSDUT when one of the following conditions is true:

- RACF is installed.
- ACF2 is installed, and the ‘SAF’ option is enabled.
- Top Secret is installed, and the ‘SAF’ option is enabled.
- The ASXBUSER field in the ASXB control block is filled in with the submitting user ID by an installation-defined method.

The SECUCMND field contains the command that caused the Security Exit program to be invoked. The available commands are:

DIR Display a directory entry for a DDIO file member (for DIRECTORY or DIRX commands).

DIRS List the directory entries in a sequential file produced by the EXPORT command.

VIEW	View the DDIO file member.
DELT	Delete the DDIO file member.
PRNT	Print the DDIO file member.
FORM	Format the DDIO file.
LOCK	Lock the DDIO file member.
UNLK	Unlock the DDIO file member.
COPY	Copy the DDIO file member.
MOVE	Move the DDIO file member.
IMPR	Import the DDIO file member.
EXPR	Export the DDIO file member.
REOR	Reorder the DDIO file.

Figure 13-1. CWASSECD DSECT

```
         MACRO
         CWASSECD
CWASSECD DSECT
SECUCMND DS    CL4    A         REQUEST BEING MADE :
*                                'DIR ' 'DIRS' 'VIEW' 'DELT'
*                                'PRNT' 'FORM' 'LOCK' 'UNLK'
*                                'COPY' 'MOVE' 'OPEN' 'PROF'
*                                'IMPR' 'EXPR' 'REOR'
*-----------------------------------------------------------------
*   AA/CICS BATCH COMMANDS:      'DELT' 'LOCK' 'UNLK' 'COPY'
*                                'MOVE' 'IMPR' 'EXPR'
*
*-----------------------------------------------------------------
         DS    CL32             RESERVED
SECUAREA DS    16F    A         SAVE AREA FOR USER SECURITY EXIT
         DS    CL132            RESERVED
SECUDSNT DS    CL44   A         DDIO DATASET NAME
SECUDSND DS    CL44   A         DESTINATION DSNAME -
*                                     COPY/MOVE/IMPORT/EXPORT ONLY
SECUSDSN DS    CL44   CO        USER ENTERED SOURCE DATASET NAME -
*                                        PRINT/VIEW/OPEN ONLY
SECUCUSR DS    CL8    A         CURRENT USER ID (EXEPT BATCH)
SECUFILE DS    X      A         TYPE OF CALL
SECUOSON EQU   X'04'               MVS/ABENDAID (ISPF)
SECUOSBT EQU   X'08'               MVS/ABENDAID (BATCH)
SECUCCON EQU   X'0C'               OLD CICS/ABENDAID (ISPF)
SECUCCDF EQU   X'10'               AA/CICS SERVER & CICS/ABENDAID (AADF)
SECUCCBT EQU   X'14'               ABENDAID FOR CICS (BATCH)
SECUSRON EQU   X'18'               SOURCE (ISPF)
SECUSRDF EQU   X'1C'               SOURCE (AA/CICS SERVER & AADF)
SECUSRBT EQU   X'20'               SOURCE (BATCH)
SECURDON EQU   X'24'               RESERVED
SECURDBT EQU   X'28'               RESERVED
         DS    0D
*
* INFORMATION SPECIFIC TO ABEND-AID
*
SECUJOB  DS    CL8    O         JOBNAME
SECUSTEP DS    CL8    O         STEPNAME
SECUPROC DS    CL8    O         PROCSTEP
SECUJES  DS    CL8    O         JES JOB ID
SECUABND DS    CL4    O         TYPE OF ABEND (ABEND CODE)
SECUTIME DS    PL4    O         TIME OF ABEND HHMMSSS+
SECUDATE DS    PL3    O         DATE OF ABEND YYDDD+
SECUPROG DS    CL10   O         PROGRAMMER NAME
         DS    CL100            RESERVED
         ORG   SECUJOB
*
         DS    CL 18            SIZE ADJUSTMENT
* INFORMATION SPECIFIC TO ABEND-AID FOR CICS AND/OR OLD CICS ABEND-AID
*
SECUTRAN DS    CL4    C         ABENDING TRANSACTION CODE
SECUTERM DS    CL4    C         TERMINAL ID OF ABENDING PGM
SECUPGRM DS    CL8    C         ABENDING PROGRAM NAME
SECUNETN DS    CL8    C         ABENDING PROGRAM VTAM NETNAME
SECUAPID DS    CL8    C         CICS APPLID WHEN ABEND TAKEN
SECUCABN DS    CL4    C         TYPE OF ABEND (ABEND CODE)
SECUCTIM DS    PL4    C         TIME OF ABEND HHMMSSS+
SECUCDAT DS    PL3    C         DATE OF ABEND YYDDD+
SECUAUSR DS    CL8    C         CICS USER ID WHEN ABEND TAKEN
SECUCPID DS    CL8    F         CURRENT CICS APPLID
SECUCTRM DS    CL4    F         CURRENT CICS TERMINAL ID
SECUPROF DS    CL8    F         USER ENTERED PROFILE KEY
         DS    CL100            RESERVED
         ORG   SECUJOB
*
* INFORMATION SPECIFIC TO LISTING FILE
*
SECUELST DS    CL8    E         PROGRAM NAME
SECUEDAT DS    CL8    E         COMPILE DATE
SECUETIM DS    CL8    E         COMPILE TIME
SECUEREL DS    CL8    E         SOURCE LEVEL RELEASE
         DS    CL100            RESERVED
*
         ORG
SECDSIZE EQU   *-CWASSECD
         MEND
```

Output Information

Upon exiting, the Security Exit program passes one of the following return codes or addresses in R15:

0 Continue processing the requested command.

4 Stop processing the requested command. The default message UNABLE TO COMPLY WITH REQUEST BECAUSE OF SECURITY REASONS is sent to the user.

ADDRESS

Stop processing the command. For CWDDSUTL commands, the customer-supplied message pointed to by the address in R15 is sent to the user instead of the address. For CWFXSDUT and CWDDLPUT commands, the customer-supplied message is sent **in addition to** the address. The customer's message must comply with the following format:

- 1 Byte: Length of the message
- 80 Bytes (maximum): Message content

Appendix A.
Using the Problem Documentation Utility

The Problem Documentation Utility collects documentation to send to Compuware Technical Support for resolving problems encountered when programs are compiled. This utility helps ensure that all required documentation for a problem is sent to Technical Support.

Sample JCL for this utility is located in the CSS sample installation library, COMPWARE.LCXnnn.SLCXCNTL.

Step 1. Add a DD Card to the Language Processor

To invoke the Problem Documentation Utility, add a CWPCAP DD card to the preprocessor or postprocessor JCL. The DD card should be coded as follows:

```
//CWPCAP DD DSN=datasetname,DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(TRK,1)
```

You can choose any dataset name that conforms to your site's installation standards. The dataset name will be used as the prefix for the documentation datasets.

Step 2. Run the Language Processor Using Your Source Program

Run the language processor using the same source program that caused the reported error. If the processor completes, several datasets will be created. The dataset name specified on the CWPCAP DD is used as the high-level qualifier for the created datasets. Table A-1 lists all possible datasets that could be created by the Problem Documentation Utility.

If the error causes the processor to abend, only some of the datasets will be created. How certain features are used may also prevent some files from being created. For example, if no COBOL copy members were used in the program, then the COPYBOOK dataset is not created.

Note: The utility does not capture language processor parameters specified on an INCLUDE parameter. Also, some files can only be captured if the preprocessor was used. These are listed in Table A-1.

Table A-1. Problem Documentation Utility Datasets

Dataset Name	Purpose
dsn.CORE.CXRELS	CSS load module containing CSS release and maintenance information.
dsn.CORE.PARMS	Language processor and/or compiler parameters specified on the EXEC.
dsn.CWDDSUTL.EXPORT	Exported DDIO member. After the postprocessor phase has completed, the DDIO member is created in a DDIO file.
dsn.FILE.FILEINFO	Describes every allocated file and its attributes. It contains the DD statement, dataset name, member name, LRECL, BLKSIZE, RECFM, and VOLSER.
dsn.IEBGENER.CWPDUMPR *	Dummy compile SYSPRINT. File CWPDUMPR contains the compiler listing from the dummy compile in order to force on required compiler options.
dsn.IEBGENER.CWPLOAD	File SYSLIN contains the object deck that can be used to re-create the customer's load module. This dataset is created provided the processor does not abend or return a bad error code.
dsn.IEBGENER.CWPPRMO	Language processor parameters
dsn.IEBGENER.CWPPRTI *	File CWPPRTI contains the compiler listing. This file is then fed into the postprocessor phase to create the SYSPRINT and the DDIO member.
dsn.IEBGENER.SYSIN *	SYSIN file fed into the compile process.
dsn.IEBUPDTE.COPYBOOK *	COBOL copybooks, PL/I Include members, or High-Level Assembler macros and copybooks. No macros or copybooks will be captured for Assembler F or H.

Note: Datasets indicated with an asterisk (*) are created by the preprocessor only.

Step 3. Copy Files to Tape

Edit sample JCL member CXCAPTUR to copy the files onto a tape. Modify this member's symbolic parameters to indicate whether any datasets are missing. For example, if no COPYBOOK dataset was created, edit the F10='DSN' parameter as follows:

```
F10='DUMMY,DSN'
```

These commands specify to the copy job not to copy a dataset to the tape. The keywords below cause the language processor to override its default values. Refer to Table A-2 for information on how to edit the symbolic parameters in CXCAPTUR.

Execute this job to copy the files and the CSS load library to a tape. Ship this tape to Compuware Technical Support, at the following address. Please include the call number, if available, on the ATTENTION line and/or the actual name of the Technical Support Representative with whom you are working.

Compuware Corporation
One Campus Martius
Detroit, MI 48226-5099
1-800-538-7822

Table A-2. Symbolic Parameters

Symbolic Parameter	Modification
UNIT=CART	Specifies the tape unit name
PREFIX='COMPWARE.CAPTURE'	Specifies the dataset name prefix that was coded on the CWPCAP DD
TPPRFX='COMPWARE.TAPE'	Specifies the dataset name prefix to be placed on the tape
TAPVOL=123456	Specifies the volume serial number of the tape
CXLOAD='COMPWARE.CX.LOAD'	Specifies the dataset name of the CSS load library
F01='DSN'	Change to F01='DUMMY,DSN' if no CORE.CXRELS dataset
F02='DSN'	Change to F02='DUMMY,DSN' if no CORE.PARMS dataset
F03='DSN'	Change to F03='DUMMY,DSN' if no CWDDSUTL.EXPORT dataset
F04='DSN'	Change to F04='DUMMY,DSN' if no FILE.FILEINFO dataset
F05='DSN'	Change to F05='DUMMY,DSN' if no IEBGENER.CWPDUMPR dataset
F06='DSN'	Change to F06='DUMMY,DSN' if no IEBGENER.CWPLOAD dataset
F07='DSN'	Change to F07='DUMMY,DSN' if no IEBGENER.CWPPRMO dataset
F08='DSN'	Change to F08='DUMMY,DSN' if no IEBGENER.CWPPRTI dataset
F09='DSN'	Change to F09='DUMMY,DSN' if no IEBGENER.SYSIN dataset
F10='DSN'	Change to F10='DUMMY,DSN' if no IEBUPDTE.COPYBOOK dataset

Appendix B.
Using the Customer Modification Facility

The Customer Modification Facility is a method for activating specialized code modifications for the CSS base code. In Pre-SMP/E releases of CSS, these changes were referred to as Restricted Zaps. If you are migrating from a release of CSS prior to CSS 7.4 and have been using one or more Restricted Zaps, you must use the Customer Modification Facility. If you have been using a Restricted Zap, you must contact Compuware Technical Support to determine the appropriate Keyword parameters to use to activate code for the corresponding Restricted Zap.

Most customers do not need to use the Customer Modification Facility.

The Customer Modification Facility reads a configuration source member containing the keywords you specify. The keywords are used to create a load module that is linked into your CSS load library. A sample of the configuration source member, and other necessary JCL, is located in the CSS sample installation library, COMPWARE.LCXnnn.SLCXCNTL. The following members are used:

- CXCFGINI - Original configuration source member
- CXCFGSET - Assembles and configures a CSS load module from the configuration source member
- CXCFGEXT - Extracts the existing configuration source member

If this is the first time you have migrated from a previous release of CSS to CSS 7.4 or more current, perform the steps in “Creating a Configuration Source Member”. If you have already set up and assembled a configuration source member and want to add additional keywords, you can choose one of the following methods:

- Add the new keywords to your copy of the CXCFGINI member. Submit CXCFGSET. For information on using CXCFGSET, refer to “Creating a Configuration Source Member”.
- Use CXCFGEXT to extract the existing source configuration member into a specified member. This member becomes the new configuration source member. The new keywords are added to the specified member and CXCFGSET must be run again. This method is used if you cannot locate your copy of the CXCFGINI member and/or don’t know which keywords have been retrofitted. For information on using CXCFGEXT for this method, refer to “Reconstructing a Configuration Source Member” on page B-3.

The CSS system default for the UNIT assignment for allocated work files can be retrofitted with the Customer Modification Facility. The Customer Modification Facility keyword is DEFUNIT and the default is SYSDA. For example, the Customer Modification Facility could be used to modify the UNIT=SYSDA default to SYSAB. For any restricted zaps other than DEFUNIT, please contact Compuware Technical Support and ask for a representative from CSS.

Creating a Configuration Source Member

Step 1. Copy member CXCFGINI

Copy member CXCFGINI from the CSS sample installation library (COMPWARE.LCXnnn.SLCXCNTL) into a member of your own dataset. Edit the copied member for the keywords you want to specify, according to the instructions in the member.

Step 2. Edit member CXCFGSET

Edit member CXCFGSET in the CSS sample installation library. CXCFGSET assembles and configures a CSS load module from member CXCFGINI. Edit the following values in this JCL:

INSRC	Dataset name that contains your configuration source member
INMEM	Name of your configuration source member
ASMBLR	Name of your assembler. Enter one of the following: IFOX00 (Assembler-F), IEV090 (Assembler-H), or ASMA90 (High Level Assembler)
USECSS	CSS load library to include in the STEPLIB concatenation
EXTCSS	CSS load library in which to build a configuration source member, usually the same as USECSS
UNIT	Temporary datasets allocated for this job.

Step 3. Submit member CXCFGSET

Submit job CXCFGSET to assemble and link-edit the configuration source member.

Reconstructing a Configuration Source Member

To reconstruct a configuration source member from a CSS load library that has been configured previously, use member CXCFGEXT. CXCFGEXT extracts a configuration source member from the existing configuration module in a CSS load library. This is the reverse of what CXCFGSET does.

Step 1. Edit Member CXCFGEXT

Edit the following values in member CXCFGEXT:

OUTPDS	Dataset name in which to build the configuration source member
OUTMEM	Member name of configuration source member to build
USECSS	CSS load library to STEPLIB
EXTCSS	CSS load library from which to read configuration data, usually the same as USECSS

Step 2. Update Extracted Member

Update the member specified in OUTMEM with any new keywords or changes required, keeping the existing keywords intact as necessary.

Step 3. Submit Member CXCFGSET

Replace the INMEM name in the CXCFGSET member with the member name specified for OUTMEM. Submit CXCFGSET. This will reconfigure the existing CSS load module to reflect the new keywords.

Appendix C.
Using the CWDDUNLP Source Extraction Utility

The CWDDUNLP utility is designed to extract source from a DDIO source listing file. Use of this utility is limited by the caution and notes supplied below. To invoke this utility, see the JCL below or reference install lib member CXDDUNLP.

```
//JOBNAME  JOB  ('XXXXXXXXXXXX'),REGION=0K,              <--- CHECK
//            CLASS=A,MSGLEVEL=(1,1),MSGCLASS=A
//*
//*****************************************************************
//*   SAMPLE JCL TO EXECUTE CWDDUNLP
//*   example JCL can also be found in install lib member (CXDDUNLP)
//*****************************************************************
//*
//STEP1    EXEC  PGM=CWDDUNLP,PARM=NAME OR NNNN           <--- CHECK
//STEPLIB  DD  DISP=SHR,DSN=COMPWARE.LCXNNN.SLCXLOAD      <--- CHECK
//SYSPRINT DD  SYSOUT=*
//SLSF0001 DD  DISP=SHR,DSN=YOUR.SOURCE.LISTING.FILE      <--- CHECK
//CWPSRC   DD  DISP=SHR,DSN=YOUR.EXTRACTED.SOURCE         <--- CHECK
//CWPCPY   DD  DSN=YOUR.EXTRACTED.COPYLIB,                <--- CHECK
//         DISP=(NEW,CATLG,DELETE),
//         DSORG=PO,LRECL=80,BLKSIZE=800,RECFM=FB,
//         SPACE=(CYL,(10,01,3),RLSE)
```

CAUTION:

The use of this utility is not intended as a replacement for keeping your original source.

Notes:

1. CWDDUNLP supports COBOL and PL/I only.
2. The CWPCPY DD is optional and works in conjunction with COBOL copybooks only.
3. CWDDUNLP runs under MVS, OS/390 platforms.
4. The output from CWDDUNLP may not be comparable to the original source.
5. Nested COBOL copy books may not appear correctly.

A parameter is passed to CWDDUNLP on the EXEC card, to specify ‘program name' or the source listing file ‘number'. If the parm passed to CWDDUNLP is numeric, the DDIO file open will be by source file number, all other opens will be by 'program name'.

Examples:

```
//STEP1  EXEC PGM=CWDDUNLP,PARM=PROGRAM1
//STEP1  EXEC PGM=CWDDUNLP,PARM=NNNN
```

The use of a **STEPLIB** is optional, but you must use the latest version of (ECC/CSS) available. This utility may not function if extracting a source member stored with a later version of (ECC/CSS) than the utility itself.

The **SYSPRINT** DD will display a small report detailing the utilities actions.

The **SLSF0001** DD statement should point to your shop's source listing file. Multiple DDIO source listing files can be specified by including additional DD statements, SLSF0002 through SLSF0006. All SLSFnnnn files are input source listing files.

The **CWPSRC DD** is a required statement. It should specify an **LRECL=80** output sequential file. It will contain the extracted source member after successful job completion.

The **CWPCPY DD** is not required (if provided, COBOL copy books will be placed as members within this PDS). If this DD is removed, all copy books are placed in stream with the CWPSRC source. If the COBOL source contains a 'COPY replacing' verb, the copy book will be placed within the CWPSRC as in stream source.

CWDDUNLP Return Codes

The following table lists common return codes from the CWDDUNLP DDIO Source Extraction Utility. Many are not user recoverable. Contact Compuware Technical Support if necessary.

Table C-1. CWDDUNLP Return Codes

Return Code	Message
0080	GETMAIN FAILURE, could not obtain needed storage space.
0081	VECTOR REQUEST BLOCK BUILD PROBLEM, check version of (ECC,CSS), or call COMPUWARE SHARED SERVICES SUPPORT.
0082	REGION VECTOR ERRORS, check version of (ECC,CSS), or call COMPUWARE SHARED SERVICES SUPPORT.
0083	ASSOCIATE VECTOR ERRORS, check version of (ECC,CSS), or call COMPUWARE SHARED SERVICES SUPPORT.
0084	SPAWN VECTOR ERRORS, check version of (ECC,CSS), or call COMPUWARE SHARED SERVICES SUPPORT.
0085	ASSOCIATE VECTOR WITH RQB ERRORS, check version of (ECC,CSS), or call COMPUWARE SHARED SERVICES SUPPORT.
0086	SYSPRINT DD PROBLEM, check LRECL=133 and compatible BLKSIZE.
0087	CWPSRC DD PROBLEM, check LRECL=80 and compatible BLKSIZE.
0088	LOAD FAILED FOR CWPIDRVB, check version of (ECC,CSS), or call COMPUWARE SHARED SERVICES SUPPORT.
0089	DDIO VERSION IS NOT EQUAL TO CWDDUNLP, check version of (ECC,CSS), or call COMPUWARE SHARED SERVICES SUPPORT.

Appendix D.
Using the CWDDCNVA Report File Conversion Utility

The CWDDCNVA utility can be used to convert an existing Abend-AID Report File to an Abend-AID Report Database format and attach the file to an existing shared directory file. This utility first modifies a record on the report file, then performs the ATTACH function, attaching it to the specified shared directory.

To invoke this utility, see the JCL in Figure D-1 or reference install lib member AACONVRT.

Figure D-1. JCL to Invoke the CWDDCNVA Utility

```
//JOBCARD  JOB  ('XXXXXXXXXX'),REGION=0K,
//          CLASS=A,MSGLEVEL=(1,1),MSGCLASS=A
//*
//*************************************************************************
//*        Sample JCL to execute CWDDCNVA
//*   Example JCL can also be found in install lib member AACONVRT
//*************************************************************************
//*
//STEP01   EXEC PGM=CWDDCNVA
//STEPLIB  DD DISP=SHR,DSN=COMPWARE.LCXNNN.SLCXLOAD         */ CSS LOADLIB
//SHRDIR   DD DISP=SHR,DSN=COMPWARE.ABEND.AID.SHRDIR        */SHARED DIRECTORY
//DATABASE DD DISP=SHR,DSN=COMPWARE.ABEND.AID.RPTFILE        */REPORT FILE
//ABNLREPT DD SYSOUT=*
//*
```

CAUTION:

This utility modifies a record on the report DDIO file, so it is advisable to make a file backup before running this utility.

Notes:

1. The SHRDIR DD statement (you must use this DDNAME) specifies an existing shared directory to which the converted report file will be attached.
2. The DATABASE DD statement (you must use this DDNAME) specifies an existing Abend-AID report DDIO file that you want to convert to the report database format. The Abend-AID report DDIO file name you specify here will remain the same for the new database format. The CWDDCNVA job merely changes the type and functioning of the DDIO file. If you want the Abend-AID report database to have a different name after it becomes attached to the shared directory, you must rename the original DDIO file prior to executing this job.
3. The STEPLIB should specify your CSS loadlib.

If you have problems after converting, and must return to the original DDIO format, you may use the following “unconvert” utility:

Using the CWDDUCNA Report File Unconvert Utility

The CWDDUCNA utility can be used to “unconvert” an Abend-AID Report File database back to its original DDIO format, and detach the report database from the its “owning” shared directory. This utility first performs the DETACH function on the report database then modifies the actual report file to reset it back to the original DDIO format.

To invoke this utility, see the JCL in Figure D-2 or reference install lib member AAUNCNVT.

Figure D-2. JCL to Invoke the CWDDUCNA Report File Unconvert Utility

```
//JOBCARD    JOB ('XXXXXXXX'),REGION=0K,
//           CLASS=A,MSGLEVEL=(1,1),MSGCLASS=A
//*
//* ***************************************************************************
//*             Sample JCL to execute CWDDUCNA
//*    Example JCL can also be found  in install lib member AAUNCNVT
//*****************************************************************************
//*
//STEP01   EXEC PGM=CWDDUCNA
//STEPLIB  DD DISP=SHR,DSN=COMPWARE.LCXNNN.SLCXLOAD     */ CSS LOADLIB
//SHRDIR   DD DISP=SHR,DSN=COMPWARE.ABEND.AID.SHRDIR */SHARED DIRECTORY
//DATABASE DD DISP=SHR,DSN=COMPWARE.ABEND.AID.RPTFILE */REPORT database
//ABNLREPT DD SYSOUT=*
//*
```

CAUTION:

This utility modifies a record on the report database file, so it is advisable to make a file backup before running this utility.

Notes:

1. The SHRDIR DD statement (you must use this DDNAME) specifies the shared directory to which the (converted) report database is attached.
2. The DATABASE DD statement (you must use this DDNAME) specifies an existing Abend-AID report database file that you want to “unconvert” back into the standard DDIO format. The Abend-AID report database file name you specify here will remain the same when it reverts back to the standard Abend-AID DDIO file format. The CWDDUNCA job merely changes the type and functioning of the Abend-AID report database back to the old style format. If you want the Abend-AID report DDIO file to have a different name after it becomes detached from the shared directory, you must rename it after executing this job.
3. The STEPLIB should specify your CSS loadlib.

Appendix E.
On-the-Fly Language Processing Support

CXOTFPRE & CXOTFPST Examples

CSS on-the-fly language processing support is performed by calling one of two REXX or CLIST members supplied by the user. Depending on the option selected by the user in the Compuware Viewing Facility when an appropriate source listing cannot be found, CXOTFPST (postprocessing REXX or CLIST) or CXOTFPRE (preprocessing REXX or CLIST) will be invoked.

To provide a complete demonstration of on-the-fly language processing support, Compuware has supplied two examples, complete with corresponding members (CLIST, PANEL, HELP, Skeleton). It is an actual working example that uses CA-Endevor as the source code and/or source listing repository.

Although the supplied sample is specific to CA-Endevor, any user-coded CLIST, REXX and/or program may be called as long as it conforms to the ISPF rules for program invocation. Refer to the following IBM ISPF manuals for more information about these APIs:

- SC34-4821-01 — *ISPF Dialog Developer's Guide and Reference*
- SC34-4819-01 — *ISPF Services Guide*

Supplied is the following:

REXX/CLIST name	Additional members	Use
CXOTFPST	P@CPST01	Language Postprocess dialog and panel
	P@HPST01	Help Panel
	CXSPST01	Skeleton Member
CXOTFPRE	P@CPRE01	Language Preprocess dialog and panel
	P@HPRE01	Help Panel
	CXSPRE01	Skeleton Member

The sample panel below (Figure E-1), is a language postprocessing example (CA-Endevor P@CPST01). A similar panel (not shown) is distributed for language preprocessing (P@CPRE01).

Figure E-1. Language Postprocessing Example for CA-Endeavor

```
Sample ----------- Retrieve Compiler Source Listing & Post-Process

Command ===>

 1.  Retrieve a Compiler source listing from Endevor
 2.  Generate JCL to Compuware LP Post-Process this Compiler source
     listing into a Compuware Source Listing format
 3.  Invoke EDIT to allow changes.  Enter SUBMIT in EDIT when complete.

ENDEVOR Information:
 ENVIRONMENT  ===> AA                 ELEMENT      ===> COMP5
 SYSTEM       ===> SAMPLE             TYPE         ===> COBSRC
 SUBSYSTEM    ===> COBOL              STAGE        ===> 1
 DATA SET NAME ===> EFHRXY0.AAOTF.COBSRC

Compuware Post-Process Information:
 STEPLIB         ===> CX.PROD.R070800.STGDPTFS.LOADLIB
 CWPDDIO         ===> EFHRXY0.LE.LST
 CWPPRMO         ===> CX.PROD.R070800.SORCMAC(CXLPCOBB)

JOBCARD Information:
 Line 1  ===> //EFHRXY0A JOB ('0AABAS9.3.2,SFA4'),'User name',
 Line 2  ===> //  CLASS=L,MSGCLASS=R,NOTIFY=EFHRXY0
```

Method of Operation

Data to be entered on these panels is presented in three parts:

1. CA-Endevor specific. Information is exactly as needed on a foreground execution of CA-Endevor.
2. Compuware Postprocess specific. Information needed to customize execution of a language postprocess.
3. JOBCARD or job specific information. Needed to submit JCL to an internal reader.

All input is “as is” in this example (that is, all data is taken exactly as entered and formatted for job execution). Since this is a sample, customers can modify any of these CLISTs, panels, etc. (see table above) to suit their individual needs.

At termination of the panel, the data entered will be:

1. Merged with a JCL skeleton.

   An ISPSLIB PDS must be available at viewing time containing the Compuware-supplied skeletons or those that the customer has modified. This PDS can be an existing ISPSLIB used site-wide and allocated at TSO logon, or it can be one that is allocated dynamically at viewer startup time. Refer to the previously mentioned ISPF manuals for more details on customizing skeletons. Refer to the section on installing the Compuware Viewing Facility in the *ECC Installation and Customization Guide* for details on dynamically allocating an ISPSLIB at Viewer startup time.
2. ISPF EDIT will be invoked to allow user changes
3. JOB submission, via use of the “SUBMIT” command in ISPF EDIT.

Refer to the documentation in the Compuware-supplied CXOTFPST CLIST for additional technical details. A similar CLIST, named CXOTFPRE, is also supplied for a language preprocess.

Glossary

Abend-AID. The Compuware product for fault diagnosis that provides an immediate, detailed analysis of application program failures in a comprehensive report format.

Abend-AID for CICS. The Compuware product for fault diagnosis that provides a full range of analysis functions for managing abnormal CICS transaction terminations (abends) and CICS region outages.

Abend-AID products. A generic name for the Abend-AID product family.

Abend-AID Report. (1) A set of records containing data extracted at abend time from the affected MVS region and associated control blocks that are stored in the Abend-AID DDIO report file. (2) A readable report produced by CWDDSUTL at view time using the records stored in the Abend-AID DDIO report file.

Abend time. The time when an abend occurs and Abend-AID products perform their analyses.

Allocation group. A set of blocks assigned to a DDIO file member when the member requires additional storage.

APF Authorization. APF can be used to restrict access to system functions and can require that all modules loaded by an authorized program be loaded from an authorized load library.All modules loaded by LMSINIT must be loaded from authorized load libraries or LMSINIT will fail. Only individuals with proper RACF authority will have permission to update authorized load libraries.

Assembler Language Processor. One of four language processors provided by Compuware. This language processor accepts Assembler output, builds sort work records, sorts and merges the records, and merges the records with the listing to produce Processor control blocks that can then be used as input to other Compuware products.

Automatic lock. Automatic locks are created whenever a member is added to a DDIO file format using the AUTODELETE=DUPS or AUTODELETE=STAGED option. The most recent member is automatically locked. See also **manual lock**.

AUTODELETE. An attribute of a DDIO file, specified during formatting of the file, that determines the action to be taken when the file becomes full and an attempt is made to add a new member to the file.

Base Services. Provides basic viewing server-related services. It enables Distributed Viewing Support.

Batch File Utility. A CSS utility used to prepare and manipulate DDIO files.

CEC. Central Electronic Complex designates the physical computer which can contain one or more LPARs. CECs can be of the older type designated as G1 through G6 or can be the new z/800 or z/900.

CEC Licensing. The number of MSUs provided by that CEC across all LPARs.

CGI. Common Gateway Interface. A way of interfacing computer programs with HTTP or WWW servers, so that a server can offer interactive sites instead of just static text and images.

CGI-bin. Common Gateway Interface-BINaries. A special directory where common gateway interface (CGI) scripts are kept.

C Language Processor. One of four language processors provided by Compuware. This language processor accepts C compiler output, builds sort work records, sorts and merges the records, and merges the records with the listing to produce processor control blocks that can then be used as input to other Compuware products.

COBOL Language Processor. One of four language processors provided by Compuware. This language processor accepts COBOL compiler output, builds sort work records, sorts and merges the records, and merges the records with the listing to produce processor control blocks that can then be used as input to other Compuware products.

Compuware Shared Services (CSS). A set of components used by several Compuware products to provide storage, retrieval, and maintenance for source listings and diagnostic reports.

Compuware Viewing Facility (Compuware/VF). The function of CWDDSUTL that allows the viewing, printing, locking, unlocking, and deleting of DDIO file members online.

Customer Modification Facility (CMF). This facility allows you to migrate from a previous release of CSS to CSS 7.4 or higher if you are using

any site-restricted zaps. This facility can retrofit your zaps to the SMP/E environment.

Customized Translation Table. An optional table provided by Compuware customers for translating non-printable characters. It is used by the batch file utility and Compuware/VF for printing or displaying lines containing storage displayed by Abend-AID products at abend time.

CWAASDUT. The batch file utility used for Abend-AID shared directories and report databases. CWAASDUT allows you to manage the files used by an Abend-AID shared directory and its associated abend report files. The shared directory is used to process all directory requests related to an abend report file and maintain information about database activity.

CWASSECU. The name of the user-coded Security Exit program.

CWDDLPUT. The batch file utility program used to manage source shared directories and databases for all of the supported languages. The shared directory maintains information about source listing processing and database activity. XPEDITER/TSO (releases 7.0, 7.1, and 7.2 only) supports source shared directory processing for C language, VisualAge PL/I, and Enterprise PL/I if using LONGNAME compiler support. XPEDITER/CICS (releases 7.4 and 7.5 as well as XPEDITER/TSO releases 7.0, 7.1, and 7.2) supports the use of source databases for all languages as standalone DDIO files only.

CWDDSUTL. The program for Compuware/VF and the batch file utility used by Abend-AID, XPEDITER/CICS, and XPEDITER/TSO. CWDDSUTL allows you to print, delete, lock, unlock, copy, move, import, and export DDIO file. members, and to list DDIO file directory entries. It also allows you to initially format a DDIO file

CWFXSDUT. The batch file utility used by Abend-AID for CICS. CWFXSDUT allows you to manage the files used by a shared directory and its associated transaction databases. The shared directory is used to process all directory requests related to a transaction database.

CWPPRMO. A language processor input dataset that contains all of the language processor options.

DBMODEL. The name of an OBJECT used to define all of the specifications needed to define a new source listing database during "dynamic database creation." The information is stored in the Source Listing Shared Directory specified. It is used to build the dynamic source listing database define and format functions when the dynamic database creation process is invoked during the C Language Processor source member create.

DDIO (Dump Dataset Input Output). A Compuware proprietary file access method. The CSS language processors analyze the output from a compiler or assembler execution and store that information in a source listing DDIO file. CSS DDIO files are used to store diagnostic reports, transaction reports, and source listings. The term **DDIO file** is a generic term used to refer to datasets that store the reports and listings from Compuware products that use CSS.

DDIO File. A generic name for a report file or source listing file.

DDIO File Member. A generic name for an diagnostic report in a report file, a source listing in a source listing file, or a transaction report in a transaction database.

Directory entry. A record in the fixed portion of the DDIO file (Directory) that contains information specific to a member. Each member has a corresponding directory entry. The number of directory entries is specified during formatting of the DDIO file and cannot be changed without reformatting the file.

DIRX Report. A report produced by the DIRX command of the batch file utility.

Distributed Viewing Support (DVS). Allows Abend-AID (version 9.0 and more recent) Abend-AID for CICS (version 4.4 and more recent) users to view both merged and base reports, where either the base report or listing file resides on a remote system. Abend-AID users can have source support across remote MVS systems that don't share DASD.

DYNCREATE. A term used to denote the "dynamic database creation" process that can be invoked when using the C Language Processor shared directory and source listing databases for its compiler output (CWPDDIO). A DBMODEL must first be defined in the desired Source Listing Shared Directory using the batch utility CWDDLPUT with DYNCREATE=YES specified to allow invoking of the dynamic database creation process. The CWPDDIO must specify the Source Listing Shared Directory DSN name. If not enough space is found in any of the attached source listing databases, and DYNCREATE=YES, then the dynamic database creation process is invoked, the new source listing database is created and attached, and the new source member is created.

Enhanced Listing. A convenient source of quick reference information and program documentation that merges DMAP and CLIST information, in

addition to error and diagnostic messages, with a COBOL source listing.

Enterprise Common Components (ECC). ECC is the packaging method for the following Compuware facilities:

- Compuware Shared Services (CSS)
- License Management System (LMS)
- Base Services
- Host Communications Interface (HCI)

Entry. A generic name for a transaction diagnostic report in a transaction database for Abend-AID for CICS.

Formatting a DDIO File. The preparation of an allocated file to be used as a DDIO file.

Full DDIO File. A DDIO file for which all allocation groups and/or all directory entries are allocated to members.

Host Communications Interface (HCI). A facility that provides connectivity between mainframe-based programmer productivity software and peer node software running on other platforms in a network. It allows application programs to communicate over any one of several protocols without knowledge of which protocol is in use at any given time.

Installation JCL Customization Facility. A CSS facility consisting of a set of screens that prompt you to enter installation information. This information is used to build the jobs necessary to perform the CSS SMP/E installation.

Language Processor (LP). A processor that converts assembler or compiler output into input for other Compuware products.

License Administration Utility (LAU). The LAU is the license administration control center for your organization's IT professional who is responsible for managing your access to Compuware products. The LAU is an ISPF application that enables

- the creation of a License File
- the import of License Certificates into a License File
- the maintenance and export of a License File
- the reporting and analysis of your License File and License Certificate information in virtual storage

Your organization's License Administrator will also set up License Management System parameters and system operation options using the LAU.

License Certificate. English-like readable electronic records that contain a portion of the information from your license agreement for a product release's use at a particular site. The License Certificate is used to update your License File.

License File. Dataset containing imported License Certificate information for all licensed releases of Compuware products. This file is in ASCII text format.

License Management System (LMS). Facility that enables you to centrally administer Compuware product License Certificates and manage access to Compuware products at your site. The LMS includes several components that together enable you to establish, maintain, diagnose, and upgrade access to the Compuware products licensed by your enterprise. The LMS replaces the customer profile utility provided with earlier versions of Compuware mainframe products.

Line command. A command entered next to the line to be processed. A line command is executed only for the specified member.

LMVERIFY. See **Verification Report**.

Locked member. A DDIO file member that was manually locked using the batch file utility or Compuware/VF, or automatically locked as a result of AUTODELETE=DUPS or AUTODELETE=STAGED processing. A manually locked member is identified by an M in the directory report or directory screen while an automatically locked member is identified by an L in the directory report or directory screen. When manually locked, a member cannot be automatically deleted when the DDIO file becomes full.

LPAR. Logical Partition is one operating system image, such as OS/390, z/OS, or z/OS-e.

LPAR Licensing. Refers to the defined capacity or "capped capacity" of the LPAR itself.

Manual lock. A member that was locked using the **L** line command in Compuware/VF or by using the LOCK batch command. Members that have been manually locked retain their locks during the autodeletion process. See also **automatic lock**.

MIPS. Millions of Instructions per Second. A unit of measure of processing performance equal to one million instructions per second.

MSU. Millions of Service Units per Hour.

Oldest member. (1) For report files, the diagnostic report with the earliest abend date and time. (2) For source listing files, the program with the earliest compile date and time.

PL/I Language Processor. One of four language processors provided by Compuware. This language processor accepts PL/I compiler output, builds sort work records and an incore symbol table of all of the identifiers, and produces Processor control blocks that can then be used as input to other Compuware products.

Primary Command. A command entered in the COMMAND INPUT field.

Problem Documentation Utility. A CSS utility that captures the documentation needed by Compuware for resolving technical issues and problems. This utility allows you to collect the documentation — such as copybooks, SYSPRINT, and DDIO members — that CSS Technical Support requires for problem resolution.

Report File. A file containing diagnostic reports that is accessed by DDIO.

Security Exit Program. A user-coded program that allows or denies execution of a requested command for a selected member by a specific user against particular DDIO files.

Server Grace. The amount of time an LPAR can run without being connected to the server.

Shared Directory. A variable-length record VSAM RRDS that maintains information about abends and language processing along with the attached database activity. A shared directory can contain Abend-AID for CICS directory records for each region and transaction dump known to a server, Abend-AID directory records for abend report processing, or source listing shared directory records necessary to process source listing database members.

SMP/E. System Modification Program/Extended is IBM's standard facility for installing and maintaining software modifications in the MVS environment.

Source Listing. A compiled listing and other information about a program stored in a source listing file.

Source Listing Database. A specially formatted source listing file, owned and managed by a shared directory, that is used by the C Language Processor for C LONGNAME support.

Source Listing File. A file containing source listings that is accessed by DDIO.

STROBE. The STROBE MVS Application Performance Measurement System is a product that determines where and how application time is spent in online regions and batch processing programs and how system resources are used. STROBE collects several types of data as it tracks activity within an MVS environment and produces a collection of reports that helps you determine where to revise applications to improve their performance.

Suppressed statements. Statements that are not displayed in the compiler listing. Suppressed statements in a DDIO source member contain an internal flag that the XPEDITER/TSO product can use to display or suppress source statements using the GEN command.

Transaction Database. A DDIO file, containing transaction reports, that is managed by a shared directory in Abend-AID for CICS. Multiple transaction reports are stored in a single transaction database.

Transaction Report. A Abend-AID for CICS report created from CICS transaction terminations that is stored in a DDIO file.

Unlocked Member. A DDIO file member that is not currently locked. The locked member can be unlocked either manually, using the batch file utility or Compuware/VF, or automatically as a result of AUTODELETE=DUPS or AUTODELETE=STAGED processing. An unlocked member in a full DDIO file formatted with AUTODELETE=YES, AUTODELETE=DUPS, or AUTODELETE=STAGED can be automatically deleted when another member is added to the file.

Verification Report (LMVERIFY). Displays the result of a verification program that checks out each product version in cache. This data is exactly what a product would receive when it runs. Running this report will prove that the License Management System environment is properly set up for the customer.

View time. The time when an diagnostic report or source listing is presented in a readable format for viewing online or printing.

Index

Special Characters

$AOFF parameter, PROCESSOR command
  COBOL Language Processor, 9-32
$ASUP parameter, PROCESSOR command
  Assembler Language Processor, 11-18
  COBOL Language Processor, 9-6, 9-32
$NG parameter, PROCESSOR command
  Assembler Language Processor, 11-18
  COBOL Language Processor, 9-5, 9-32
  PL/I Language Processor, 10-21
$NIDR parameter, PROCESSOR command
  Assembler Language Processor, 11-19
  C Language Processor, 12-16
  COBOL Language Processor, 9-33
  PL/I Language Processor, 10-21
$NO parameter, PROCESSOR command
  Assembler Language Processor, 11-19
  C Language Processor, 12-16
  COBOL Language Processor, 9-5, 9-33
  PL/I Language Processor, 10-21

A

AACONVRT, 6-1
AAREPORT option
  Compuware/VF, 8-8
  CSS Utilities, 3-37
ABEND parameter
  CWDDSUTL batch utility, 4-6
abend time, G-1
Abend-AID, G-1
  displaying report file information, 3-46
  executing, 1-7
  primary menu, 8-1
  setting report file options, 3-37
Abend-AID for CICS, G-1
  executing, 1-7
  Security Exit considerations, 13-2
  Security Exit Program report files, 13-2
Abend-AID products, G-1
Abend-AID report DDIO files, 3-1
  advanced parameters, 3-52
  copying members, 3-47–3-48
  creating, 3-38
  Data Management Control Blocks section, 8-26
  deleting, 3-45
  description of, 2-2
  displaying, 8-21
  displaying file information, 3-46
  displaying members, 3-48
  exporting members, 3-50
  formatting, 3-38
  importing members, 3-51
  members, 3-47
  moving members, 3-50
  Program Listing section, 8-25
  Program Storage section, 8-23
  reformatting, 3-38
  Security Exit Program, 13-2
  selecting members, 3-52
Abend-AID report file DIRX report, 2-24
ABNLFROM DD statement
  CWAASDUT batch utility, 6-3
  CWDDLPUT batch utility, 7-4
  CWDDSUTL batch utility, 4-3
  CWFXSDUT batch utility, 5-3
ABNLPARM DD statement
  CWAASDUT batch utility, 6-3
  CWDDLPUT batch utility, 7-3
  CWDDSUTL batch utility, 4-3
  CWFXSDUT batch utility, 5-3
ABNLREPT DD statement
  CWAASDUT batch utility, 6-3
  CWDDLPUT batch utility, 7-3
  CWDDSUTL batch utility, 4-4
  CWFXSDUT batch utility, 5-3
ABNLTO DD statement
  CWAASDUT batch utility, 6-4
  CWDDLPUT batch utility, 7-4
  CWDDSUTL batch utility, 4-3
  CWFXSDUT batch utility, 5-3
ABNLTO DD statement for the EXPORT command
  CWDDSUTL batch utility, 4-3
Acrobat PDF online documentation, xxiv
add a DD card to the Language Processor, A-1
advanced create/format parameters, 3-15
allocating
  allocating and formatting a new shared directory, 7-23
  DDIO files, 2-1
  VSAM cluster, 2-13
allocating and formatting a new Source Listing Database, 7-24
allocation group, G-1
alternative reserved word tables, COBOL, 9-5
AM parameter
  CWAASDUT batch utility, 6-5
  CWDDLPUT batch utility, 7-6
  CWFXSDUT batch utility, 5-5
anyname1 DD statement
  CWDDLPUT batch utility, 7-4
  CWDDSUTL batch utility, 4-4
APF authorization
  definition, G-1
APPLID parameter
  CWFXSDUT batch utility, 5-6
ASMBLR, B-2
ASSEMBLER command
  Assembler Language Processor, 11-3
Assembler Language Processor, G-1
  ASSEMBLER command, 11-3, 11-7
  assemblers, 1-2
  CONDDDIO command, 11-3, 11-8
  CONDSTEP command, 11-3, 11-8
  DDIO command, 11-3, 11-9
  DECK File-overrides command, 11-12
  description of, 11-1

ERRM File-overrides command, 11-12
File-overrides command, 11-3
File-overrides commands, 11-11
File-overrides, optional override values, 11-12
handling suppressed statements, 11-5
INCLUDE command, 11-3, 11-14
insufficient primary space allocation, 11-5
LANGPARM command, 11-3, 11-15
LANGUAGE command, 11-3, 11-15
LOAD File-overrides command, 11-12
PRINT command, 11-3, 11-16
PRINT NOGEN, 1-5
PRINT OFF, 1-5
processing listings with errors, 11-5
PROCESSOR command, 11-3, 11-17
PROGRAM-NAME command, 11-3, 11-19
PRTI File-overrides command, 11-11
PRTO File-overrides command, 11-11
required assembler options, 11-6
sample JCL and program names, 11-4
SYSTEM command, 11-3, 11-20
usage considerations, 11-5
usage restrictions, 11-6
WRK0-WRK4 File-overrides commands, 11-11
ATTACH command
CWAASDUT batch utility, 6-2, 6-14
CWDDLPUT batch utility, 7-1, 7-17
CWFXSDUT batch utility, 5-1, 5-15
audience, xxiv
AUTODELETE, G-1
AUTODELETE parameter, 2-5, 2-7, 2-20, 2-23, 2-27–2-28, 2-33, 2-35, 2-52, 2-55
automatic and manual locking, 2-7
CSS Utilities, 3-19, 3-45
CWAASDUT batch utility, 6-5
CWDDLPUT batch utility, 7-6
CWDDSUTL batch utility, 4-6
CWFXSDUT batch utility, 5-6
resolving a file full condition, 2-8
AUTODELETE=DUPS, 3-19, 3-45, 7-7
AUTODELETE=NO, 3-19, 7-8
AUTODELETE=STAGED, 2-14, 2-23, 3-19
AUTODELETE=YES, 3-19, 3-45, 7-6
automated compiler options, 1-6
automatic lock, G-1

B

BASEDDIO (BDDIO) parameter
CWDDSUTL batch utility, 4-8
BASENUMBER parameter
CWDDSUTL batch utility, 4-8
batch compile of multiple C programs in a PDS, 12-5
batch file utilities, 1-3
batch file utility
definition, G-1
batch utilities, CWAASDUT
ATTACH command, 6-2, 6-14
CHANGE command, 6-2, 6-15
command conventions, 6-4
command summary, 6-2
COPY command, 6-2, 6-16
CREATE command, 6-2, 6-17
DD statements, 6-3
DELETE command, 6-2, 6-20
DETACH command, 6-2, 6-22
DIRECTORY command, 6-2, 6-23
DIRS command, 6-2, 6-24
DIRX command, 6-2, 6-23
ESTIMATE command, 6-2, 6-25
EXAMINE command, 6-2, 6-26
EXPAND command, 6-2, 6-27
EXPORT command, 6-2, 6-28
IMPORT command, 6-2, 6-30
LOCK command, 6-2, 6-31
MOVE command, 6-2, 6-32
RESET command, 6-2, 6-33
SCRATCH command, 6-3, 6-34
UNLOCK command, 6-3, 6-35
usage considerations, 6-3
VERIFY command, 6-3, 6-36
batch utilities, CWDDLPUT
ATTACH command, 7-1, 7-17
CHANGE command, 7-2, 7-18
command conventions, 7-5
command summary, 7-1
CONVERT command, 7-2, 7-20
COPY command, 7-2, 7-21
CREATE command, 7-2, 7-22
DD statements, 7-3
DELETE command, 7-2, 7-27
description of, 2-4
DETACH command, 7-2, 7-30
DIRECTORY command, 7-2, 7-31
DIRS command, 7-2, 7-32
DIRX command, 7-2, 7-31
ESTIMATE command, 7-2, 7-33
EXPAND command, 7-2, 7-34
EXPORT command, 7-2, 7-35
IMPORT command, 7-2, 7-37
LOCK command, 7-2, 7-38
MOVE command, 7-2, 7-39
PRINT command, 7-2, 7-40
RESET command, 7-2, 7-41
SCRATCH command, 7-2, 7-41
TEST command, 7-2, 7-42
UNCONVERT command, 7-3, 7-43
UNLOCK command, 7-3, 7-44
usage considerations, 7-3
VERIFY command, 7-3, 7-45
batch utilities, CWDDSUTL
command conventions, 4-5
command summary, 4-1
COPY command, 4-1, 4-15
DELETE command, 4-1, 4-16
description of, 2-4
DIRECTORY command, 4-1, 4-17
DIRS command, 4-1, 4-18
DIRX command, 4-2, 4-17
EXPORT command, 4-2, 4-19
FLAG command, 4-2, 4-20
FORMAT command, 4-2, 4-23
IMPORT command, 4-2, 4-25
LOCK command, 4-2, 4-26
MOVE command, 4-2, 4-27
PRINT command, 4-2, 4-28
REORDER command, 4-2, 4-29
UNLOCK command, 4-2, 4-30
batch utilities, CWFXSDUT
ATTACH command, 5-1, 5-15
CHANGE command, 5-1, 5-16
command conventions, 5-5
command summary, 5-1
COPY command, 5-1, 5-17

CREATE command, 5-1, 5-18
DD statements, 5-3
DELETE command, 5-1, 5-22
description of, 2-4
DETACH command, 5-2, 5-24
DIRECTORY command, 5-2, 5-25
DIRS command, 5-2, 5-26
DIRX command, 5-2, 5-25
ESTIMATE command, 5-2, 5-27
EXAMINE command, 5-2, 5-28
EXPAND command, 5-2, 5-30
EXPORT command, 5-2, 5-31
IMPORT command, 5-2, 5-33
LOCK command, 5-2, 5-34
MOVE command, 5-2, 5-35
RESET command, 5-2, 5-36
SCRATCH command, 5-2, 5-36
UNLOCK command, 5-2, 5-37
usage considerations, 5-3
VERIFY command, 5-2, 5-38
benefits of CSS, 1-1
BLKSIZE parameter, 2-5
CWAASDUT batch utility, 6-7
CWDDLPUT batch utility, 7-9
CWDDSUTL batch utility, 4-8
CWFXSDUT batch utility, 5-7
BookManager softcopy documentation, xxiv
BYPASS parameter, PROCESSOR command
Assembler Language Processor, 11-17
COBOL Language Processor, 9-28
PL/I Language Processor, 10-20

C

C Language Processor, G-1
batch compile of multiple C programs in a PDS, 12-5
C command, 12-3, 12-8
compilers, 1-2
CONDDDIO command, 12-3, 12-9
CONDSTEP command, 12-3, 12-9
DD statements, 12-4
DDIO command, 12-3, 12-10
description of, 12-1
effect of optimization logic on execution pointer, 12-5
ERRM File-overrides commands, 12-11
File-overrides command, 12-3
File-overrides commands, 12-11
HFS files, use of, 12-5
INCLUDE command, 12-3, 12-13
insufficient primary space allocation, 12-5
LANGPARM command, 12-3, 12-14
LANGUAGE command, 12-3, 12-14
LOAD File-overrides command, 12-12
LONGNAME support, 2-3
NOSHOWINC, 1-5
OPTFILE compiler option, 12-5
PRINT command, 12-3, 12-15
processing listings with errors, 12-5
PROCESSOR command, 12-3, 12-16
PROGRAM-NAME command, 12-3, 12-17
PRTI File-overrides command, 12-11
PRTO File-overrides command, 12-11
required compiler options, 12-7
restrictions, 12-6
sample JCL and program names, 12-4
SYSTEM command, 12-3, 12-17
usage considerations, 12-5
calculate
CALCULATE synonym, 9-5
DDIO file size, 2-13
DDIO files, allocation in blocks, 2-13
DDIO files, minimum value for BLKSIZE, 2-11
DDIO files, minimum value for CISIZE, 2-11, 2-13
DDIO files, total number of blocks, 2-11, 2-13
DDIO files, value for GROUPCOUNT, 2-11
value for GROUPCOUNT, 2-13
capturing suppressed source code, preprocessor, 1-5
CEC, G-1
licensing, G-1
CGI, G-1
CHANGE command
CWAASDUT batch utility, 6-2, 6-15
CWDDLPUT batch utility, 7-2, 7-18
CWFXSDUT batch utility, 5-1, 5-16
changes, summary of, xix
CICSCODE parameter
CWFXSDUT batch utility, 5-7
CICSTEST command
COBOL Language Processor, 9-3, 9-13
CISIZE parameter
calculate minimum value, 2-13
CWAASDUT batch utility, 6-7
CWDDLPUT batch utility, 7-9
CWFXSDUT batch utility, 5-7
CMF (Customer Modification Facility), B-1
COBOL Language Processor, G-1
alternative reserved word tables, 9-5
CICSTEST command, 9-3, 9-13
COBOL command, 9-3, 9-14
command descriptions, 9-13
compilers, 1-2
CONDDDIO command, 9-3, 9-15
CONDERRL command, 9-3, 9-15
CONDSTEP command, 9-3, 9-16
COPY SUPPRESS, 1-5
CWPPRMO, 9-4
DD statements, 9-4
DDIO command, 9-3, 9-17
DECK File-overrides command, 9-20
description of, 9-1
effect of optimization logic on execution pointer, 9-6
enhanced listing, 9-10
enhanced listing features, 9-12
ERRM File-overrides command, 9-20
File-overrides commands, 9-3, 9-19
File-overrides, optional override values, 9-20
INCLUDE command, 9-3, 9-22
insufficient primary space allocation, 9-6
LANGPARM command, 9-3, 9-23
LANGUAGE command, 9-3, 9-24
LOAD File-overrides command, 9-20
output considerations, 9-5
performance considerations, 9-7
preventing truncated messages, 9-5
PRINT command, 9-3, 9-26
processing listings with errors, 9-5
PROCESSOR command, 9-3, 9-28
PROGRAM-NAME command, 9-3, 9-34
PRTI File-overrides command, 9-19
PRTO File-overrides command, 9-19
required compiler options, 9-9

restricted compiler options, 9-7
restrictions, 9-7
sample JCL and program names, 9-4
suppressed statements, handling, 9-6
SYSTEM command, 9-3, 9-34
usage considerations, 9-4–9-5
WRK0-WRK4 File-overrides command, 9-19
command conventions
CWAASDUT batch utility, 6-4
CWDDLPUT batch utility, 7-5
CWDDSUTL batch utility, 4-5
CWFXSDUT batch utility, 5-5
command syntax
Assembler Language Processor, 11-7
C Language Processor, 12-8
COBOL Language Processor, 9-13
CWAASDUT batch utility, 6-14
CWDDLPUT batch utility, 7-17
CWDDSUTL batch utility, 4-15
CWFXSDUT batch utility, 5-15
PL/I Language Processor, 10-11
commands
existing DDIO files, 4-15
commands, Assembler Language Processor
ASSEMBLER, 11-3, 11-7
CONDDDIO, 11-3, 11-8
CONDSTEP, 11-3, 11-8
DDIO, 11-3, 11-9
DECK File-overrides, 11-12
ERRM File-overrides, 11-12
File-overrides, 11-3, 11-11
File-overrides, optional override values, 11-12
INCLUDE, 11-3, 11-14
LANGPARM, 11-3, 11-15
LANGUAGE, 11-3, 11-15
LOAD File-overrides, 11-12
PRINT, 11-3, 11-16
PROCESSOR, 11-3, 11-17
PROGRAM-NAME, 11-3, 11-19
PRTI File-overrides, 11-11
PRTO File-overrides, 11-11
SYSTEM, 11-3, 11-20
WRK0-WRK4 File-overrides, 11-11
commands, C Language Processor
C, 12-3, 12-8
CONDDDIO, 12-3, 12-9
CONDSTEP, 12-3, 12-9
DDIO, 12-3, 12-10
ERRM File-overrides, 12-11
File-overrides, 12-3, 12-11
INCLUDE, 12-3, 12-13
LANGPARM, 12-3, 12-14
LANGUAGE, 12-3, 12-14
LOAD File-overrides, 12-12
PRINT, 12-3, 12-15
PROCESSOR, 12-3, 12-16
PROGRAM-NAME, 12-3, 12-17
PRTI File-overrides, 12-11
PRTO File-overrides, 12-11
SYSTEM, 12-3, 12-17
commands, COBOL Language Processor
CICSTEST, 9-3, 9-13
COBOL, 9-3, 9-14
CONDDDIO, 9-3, 9-15
CONDERRL, 9-3, 9-15
CONDSTEP, 9-3, 9-16
DDIO, 9-3, 9-17
DECK File-overrides, 9-20
ERRM File-overrides, 9-20
File-overrides, 9-3, 9-19
File-overrides, optional override values, 9-20
INCLUDE, 9-3, 9-22
LANGPARM, 9-3, 9-23
LANGUAGE, 9-3, 9-24
LOAD File-overrides, 9-20
PRINT, 9-3, 9-26
PROCESSOR, 9-3, 9-28
PROGRAM-NAME, 9-3, 9-34
PRTI File-overrides, 9-19
PRTO File-overrides, 9-19
SYSTEM, 9-3, 9-34
WRK0-WRK4 File-overrides, 9-19
commands, CWAASDUT batch utility
ATTACH, 6-2, 6-14
CHANGE, 6-2, 6-15
COPY, 6-2, 6-16
CREATE, 6-2, 6-17
DELETE, 6-2, 6-20
DETACH, 6-2, 6-22
DIRECTORY, 6-2, 6-23
DIRS, 6-2, 6-24
DIRX, 6-2, 6-23
ESTIMATE, 6-2, 6-25
EXAMINE, 6-2, 6-26
EXPAND, 6-2, 6-27
EXPORT, 6-2, 6-28
IMPORT, 6-2, 6-30
LOCK, 6-2, 6-31
MOVE, 6-2, 6-32
RESET, 6-2, 6-33
SCRATCH, 6-3, 6-34
UNLOCK, 6-3, 6-35
VERIFY, 6-3, 6-36
commands, CWDDLPUT batch utility
ATTACH, 7-1, 7-17
CHANGE, 7-2, 7-18
CONVERT, 7-2, 7-20
COPY, 7-2, 7-21
CREATE, 7-2, 7-22
DELETE, 7-2, 7-27
DETACH, 7-2, 7-30
DIRECTORY, 7-2, 7-31
DIRS, 7-2, 7-32
DIRX, 7-2, 7-31
ESTIMATE, 7-2, 7-33
EXPAND, 7-2, 7-34
EXPORT, 7-2, 7-35
IMPORT, 7-2, 7-37
LOCK, 7-2, 7-38
MOVE, 7-2, 7-39
PRINT, 7-2, 7-40
RESET, 7-2, 7-41
SCRATCH, 7-2, 7-41
TEST, 7-2, 7-42
UNCONVERT, 7-3, 7-43
UNLOCK, 7-3, 7-44
VERIFY, 7-3, 7-45
commands, CWDDSUTL batch utility, 4-15
COPY, 4-1, 4-15
DELETE, 4-1, 4-16
DIRECTORY, 4-1, 4-17
DIRS, 4-1, 4-18
DIRX, 4-2, 4-17
EXPORT, 4-2, 4-19
FLAG, 4-2, 4-20
FORMAT, 4-2, 4-23

IMPORT, 4-2, 4-25
LOCK, 4-2, 4-26
MOVE, 4-2, 4-27
PRINT, 4-2, 4-28
REORDER, 4-2, 4-29
UNLOCK, 4-2, 4-30
commands, CWFXSDUT batch utility
ATTACH, 5-1, 5-15
CHANGE, 5-1, 5-16
COPY, 5-1, 5-17
CREATE, 5-1, 5-18
DELETE, 5-1, 5-22
DETACH, 5-24
DIRECTORY, 5-2, 5-25
DIRS, 5-2, 5-26
DIRX, 5-2, 5-25
ESTIMATE, 5-2, 5-27
EXAMINE, 5-2, 5-28
EXPAND, 5-2, 5-30
EXPORT, 5-2, 5-31
IMPORT, 5-2, 5-33
LOCK, 5-2, 5-34
MOVE, 5-2, 5-35
RESET, 5-2, 5-36
SCRATCH, 5-2, 5-36
UNLOCK, 5-2, 5-37
VERIFY, 5-2, 5-38
commands, PL/I Language Processor
CONDDDIO, 10-3, 10-11
CONDSTEP, 10-3, 10-11
DDIO, 10-3, 10-12
DECK File-overrides command, 10-14
ERRM File-overrides, 10-14
File-overrides, 10-3, 10-13
File-overrides, optional override values, 10-14
INCLUDE, 10-3, 10-16
LANGPARM, 10-3, 10-17
LANGUAGE, 10-3, 10-17
LOAD File-overrides command, 10-14
PL/I, 10-3, 10-18
PRINT, 10-3, 10-19
PROCESSOR, 10-3, 10-20
PROGRAM-NAME, 10-3, 10-22
PRTI File-overrides, 10-13
PRTO File-overrides, 10-14
SYSTEM, 10-3, 10-22
WRK0-WRK4 File-overrides, 10-13
compile errors, handling
preprocessor, 1-5
compilers
C language processor, 1-2
COBOL language processor, 1-2
PL/I language processor, 1-2
components of CSS, 1-3
COMPUTE synonym, 9-5
Compuware Shared Services
definition, G-1
Compuware/VF
AAREPORT option, 8-8
Abend-AID primary menu, 8-1
Abend-AID report files, viewing, 8-21
accessing CSS Utilities, 8-44
accessing File-AID, 8-27, 8-44
contact information, 8-9
creating JCL for language processing, 8-7
data locator, 8-28
dataset directory screen, 8-10
definition, G-1
entry panel, 8-2
LOCATE command, 8-16
NEXT command, 8-17
Output Selection menu, 8-21
preprocessing and postprocessing, 8-7
primary commands, 8-16
required parameters, 8-7
SORT command, 8-17
source listing files, viewing, 8-21
SOURCE option, 8-7
VIEW option, 8-2
Compuware's Frontline web site, xxvii
Compuware's phone number and address, xxvii
CONDDDIO command
Assembler Language Processor, 11-3, 11-8
C Language Processor, 12-3, 12-9
COBOL Language Processor, 9-3, 9-5, 9-15
PL/I Language Processor, 10-3, 10-11
CONDERRL command
COBOL Language Processor, 9-3, 9-15
CONDSTEP command
Assembler Language Processor, 11-3, 11-8
C Language Processor, 12-3, 12-9
COBOL Language Processor, 9-3, 9-16
PL/I Language Processor, 10-3, 10-11
configuration source members
create, B-2
CXCFGINI, B-2
CXCFGSET, B-2
Contact Information
displaying, 8-10
contact information, 8-9
maintaining, 8-9
CONTINUE parameter
CWAASDUT batch utility, 6-7
CWDDLPUT batch utility, 7-9
CWDDSUTL batch utility, 4-8
CWFXSDUT batch utility, 5-7
conventions, xxv
CONVERT command
CWDDLPUT batch utility, 7-2, 7-20
COPY command
CWAASDUT batch utility, 6-2, 6-16
CWDDLPUT batch utility, 7-2, 7-21
CWDDSUTL batch utility, 4-1, 4-15
CWFXSDUT batch utility, 5-1, 5-17
copy files to tape, A-3
copying
Abend-AID report file members, 3-48
source listing file members, 3-24
CREATE command
CWAASDUT batch utility, 6-2, 6-17
CWDDLPUT batch utility, 7-2, 7-22
CWFXSDUT batch utility, 5-1, 5-18
CREATEJCL option
Compuware/VF, 8-7
CSS Utilities, 3-4
creating
Abend-AID report DDIO files, 3-38
DBMODEL, 7-26
Source Listing Databases, CWDDLPUT batch utility, 7-3
source listing DDIO files, 3-12
CSS
allocating and formatting DDIO files, 2-1
benefits, 1-1
components, 1-3
executing, 1-7

overview of, 1-1
release level, viewing, 8-6
Security Exit Program, 13-1
CSS Utilities, 1-3
AAREPORT option, 3-37
Abend-AID report files, working with, 3-37
accessing from Compuware/VF, 8-44
CREATEJCL option, 3-4
description of, 2-3
functions of, 3-1
global settings, 3-3
invoking, 3-2
postprocessor, 3-8
preparing compiler listings, 3-4
preprocessor, 3-5
source listing files, working with, 3-10
SOURCE option, 3-10
CURRDDIO parameter
CWDDSUTL batch utility, 4-8
CURRNUMBER parameter
CWDDSUTL batch utility, 4-8
Customer Modification Facility, B-1
definition, G-1
customer support web site, xxiv
customized translation table, G-2
CWAASDUT, G-2
CWAASDUT batch utility
ATTACH command, 6-2, 6-14
CHANGE command, 6-2, 6-15
command conventions, 6-4
command summary, 6-2
COPY command, 6-2, 6-16
CREATE command, 6-2, 6-17
DD statements, ABNLFROM, 6-3
DD statements, ABNLPARM, 6-3
DD statements, ABNLREPT, 6-3
DD statements, ABNLTO, 6-4
DELETE command, 6-2, 6-20
DETACH command, 6-2, 6-22
DIRECTORY command, 6-2, 6-23
DIRS command, 6-2, 6-24
DIRX command, 6-2, 6-23
ESTIMATE command, 6-2, 6-25
EXAMINE command, 6-2, 6-26
EXPAND command, 6-2, 6-27
EXPORT command, 6-2, 6-28
IMPORT command, 6-2, 6-30
LOCK command, 6-2, 6-31
MOVE command, 6-2, 6-32
parameters, 6-5
renaming report files, 6-3
renaming shared directories, 6-3
RESET command, 6-2, 6-33
SCRATCH command, 6-3, 6-34
UNLOCK command, 6-3, 6-35
usage considerations, 6-3
usage restrictions, 6-4
VERIFY command, 6-3, 6-36
CWASSECU, G-2
CWDDLPUT, G-2
CWDDLPUT batch utility
ABNLFROM DD statement, 7-4
ABNLPARM DD statement, 7-3
ABNLREPT DD statement, 7-3
ABNLTO DD statement, 7-4
allocating and formatting a new shared directory, 7-23
allocating and formatting a new Source Listing Database, 7-24
anyname1 DD statement, 7-4
ATTACH command, 7-1, 7-17
CHANGE command, 7-2, 7-18
command conventions, 7-5
command summary, 7-1
CONVERT command, 7-20
CONVERT convert, 7-2
COPY command, 7-2, 7-21
CREATE command, 7-2, 7-22
creating a DBMODEL, 7-26
creating Source Listing Databases, 7-3
DD statements, 7-3
DELETE command, 7-2, 7-27
description of, 2-4
DETACH command, 7-2, 7-30
DIRECTORY command, 7-2, 7-31
DIRS command, 7-2, 7-32
DIRX command, 7-2, 7-31
ESTIMATE command, 7-2, 7-33
EXPAND command, 7-2, 7-34
EXPORT command, 7-2, 7-35
formatting a pre-allocated Shared Directory, 7-23
formatting a pre-allocated Source Listing Database, 7-25
FROMDD parameter, 7-11
IMPORT command, 7-2, 7-37
LOCK command, 7-2, 7-38
MOVE command, 7-2, 7-39
parameters, 7-6
PRINT command, 7-2, 7-40
renaming shared directories, 7-3
renaming transaction databases, 7-3
RESET command, 7-2, 7-41
SCRATCH command, 7-2, 7-41
TEST command, 7-2, 7-42
UNCONVERT command, 7-43
UNCONVERT convert, 7-3
UNLOCK command, 7-3, 7-44
usage considerations, 7-3
usage restrictions, 7-5
VERIFY command, 7-3, 7-45
CWDDSUTL, G-2
CWDDSUTL batch utility
command conventions, 4-5
command summary, 4-1
COPY command, 4-1, 4-15
DD statement, ABNDFROM, 4-3
DD statement, ABNLPARM, 4-3
DD statement, ABNLREPT, 4-4
DD statement, ABNLTO, 4-3
DD statement, ABNLTO for the EXPORT command, 4-3
DD statement, anyname1, 4-4
DD statement, SLSF001, 4-3
DELETE command, 4-1, 4-16
description of, 2-4
DIRECTORY command, 4-1, 4-17
DIRS command, 4-1, 4-18
DIRX command, 4-2, 4-17
EXPORT command, 4-2, 4-19
FLAG command, 4-2, 4-20
FORMAT command, 4-2, 4-23
IMPORT command, 4-2, 4-25
LOCK command, 4-2, 4-26
MOVE command, 4-2, 4-27
parameters, 4-6

PRINT command, 4-2, 4-28
REORDER command, 4-2, 4-29
UNLOCK command, 4-2, 4-30
usage considerations, 4-2
usage restrictions, 4-4
CWDDUNLP source extraction utility, 2-4
return codes, C-2
CWFXSDUT, G-2
CWFXSDUT batch utility
ATTACH command, 5-1, 5-15
CHANGE command, 5-1, 5-16
command summary, 5-1
COPY command, 5-1, 5-17
CREATE command, 5-1, 5-18
DD statements, ABNLFROM, 5-3
DD statements, ABNLPARM, 5-3
DD statements, ABNLREPT, 5-3
DD statements, ABNLTO, 5-3
DD statements, FDBDCUST, 5-3
DELETE command, 5-1, 5-22
description of, 2-4
DETACH command, 5-2, 5-24
DIRECTORY command, 5-2, 5-25
DIRS command, 5-2, 5-26
DIRX command, 5-2, 5-25
ESTIMATE command, 5-2, 5-27
EXAMINE command, 5-2, 5-28
EXPAND command, 5-2, 5-30
EXPORT command, 5-2, 5-31
IMPORT command, 5-2, 5-33
LOCK command, 5-2, 5-34
MOVE command, 5-2, 5-35
parameters, 5-5
renaming shared directories, 5-3
renaming transaction databases, 5-3
RESET command, 5-2, 5-36
SCRATCH command, 5-2, 5-36
UNLOCK command, 5-2, 5-37
usage considerations, 5-3
usage restrictions, 5-4
VERIFY command, 5-2, 5-38
CWPCMAIN, 9-5
CWPPRMO, G-2
CWPPRMO DD statement, 9-5
CWPPRTI DD statement
preprocessor, COBOL, 9-6
CWPPRTO
COBOL Language Processor, 9-5
CXCFGEXT member
edit, B-3
CXCFGINI
configuration source member, B-2
CXCFGSET
configuration source member, B-2
edit member, B-2
submit member, B-2–B-3
CYLINDERS parameter
CWAASDUT batch utility, 6-7
CWDDLPUT batch utility, 7-9
CWFXSDUT batch utility, 5-8
CYNCREATE, G-2

D

data locator (Compuware/VF), 8-28
creating reports, 8-28
description of, 8-29
printing reports, 8-30
Data Management Control Blocks section, 8-26
DATABASEDISP parameter
CWAASDUT batch utility, 6-7
CWDDLPUT batch utility, 7-9
CWFXSDUT batch utility, 5-8
DATACLAS parameter
CWAASDUT batch utility, 6-8
CWDDLPUT batch utility, 7-9
CWFXSDUT batch utility, 5-8
dataset directory screen, 8-10
Abend-AID report file, 8-10
line commands, 8-18
source listing file, 8-12
DATE parameter
DDIO command, 11-9
DAYSOLD parameter
CWAASDUT batch utility, 6-8
CWDDLPUT batch utility, 7-9
CWDDSUTL batch utility, 4-9
CWFXSDUT batch utility, 5-8
DBDISP parameter
CWAASDUT batch utility, 6-7
CWFXSDUT batch utility, 5-8
DBMODEL, G-2
DD statements, Assembler Language Processor, 11-4
DD statements, C Language Processor, 12-4
DD statements, COBOL Language Processor, 9-4
DD statements, CWAASDUT batch utility, 6-3
ABNLFROM, 6-3
ABNLPARM, 6-3
ABNLREPT, 6-3
ABNLTO, 6-4
DD statements, CWDDLPUT batch utility, 7-3
ABNLPARM, 7-3
ABNLREPT, 7-3
ABNLTO, 7-4
ANBLFROM, 7-4
anyname1, 7-4
DD statements, CWDDSUTL batch utility, 4-2
ABNLFROM, 4-3
ABNLPARM, 4-3
ABNLREPT, 4-4
ABNLTO, 4-3
anyname1, 4-4
SLSF001, 4-3
DD statements, CWFXSDUT batch utility, 5-3
ABNLFROM, 5-3
ABNLPARM, 5-3
ABNLREPT, 5-3
ABNLTO, 5-3
FDBDCUST, 5-3
DD statements, PL/I Language Processor, 10-4
DDIO, G-2
DDIO calculator, 2-15, 3-61
helpful hints, 3-63
primary calculations screen, 3-61
DDIO command
Assembler Language Processor, 11-3, 11-9
C Language Processor, 12-3, 12-10
COBOL Language Processor, 9-3, 9-17
PL/I Language Processor, 10-3, 10-12
DDIO file, G-2
DDIO file member, G-2
DDIO files
Abend-AID report, 2-2
allocating, 2-9

allocating and formatting, 2-1
allocating and formatting example, 2-13
allocating the VSAM cluster, 2-13
AUTODELETE parameter, 2-5
BLKSIZE parameter, 2-5
calculate allocation in blocks, 2-10, 2-13
calculate average member size in KBs, 2-9
calculate file size, 2-10, 2-13
calculate minimum value for CISIZE, 2-13
calculate total number of blocks, 2-11, 2-13
calculate value for BLKSIZE, 2-11
calculate value for CISIZE, 2-11
calculate value for GROUPCOUNT, 2-11, 2-13
decreasing CISIZE, 2-15
definition, xxv
description of, 1-3
determine average member size, 2-13
EXTENTS parameter, 2-5
file considerations, 2-7
file full, 5-7, 6-6, 7-8
file full condition, 2-8, 4-7
formatting, 2-10, 2-13
formatting Abend-AID report DDIO files, 3-38
formatting formulas, 2-11
formatting specifications, 2-14
functions, 3-1
GROUPCOUNT parameter, 2-5
INDEX parameter, 2-5
limitations, 2-12
locking members, 2-7
parameters for allocating and formatting, 2-5
reducing the size of, 2-16
reformatting Abend-AID report DDIO files, 3-38
REPTCOUNT parameter, 2-6
resolving file full condition, 2-8
selecting a file access method, 2-6
shared directories, 2-3
size comparison table, 2-16
source listing, 2-2
transaction database, 2-3
tuning, 2-15
viewing file information, 2-4
DDIO member name, 12-4
DDIOCALC DIRX report data input screen, 3-62
DDNAME parameter
CWAASDUT batch utility, 6-8
CWDDLPUT batch utility, 7-10
CWFXSDUT batch utility, 5-8
DECK File-overrides command
Assembler Language Processor, 11-12
COBOL Language Processor, 9-20
PL/I Language Processor, 10-14
DEFUNIT keyword, B-1
DELETE command
CWAASDUT batch utility, 6-2, 6-20
CWDDLPUT batch utility, 7-2
CWDDSUTL batch utility, 4-1, 4-16
CWFXSDUT batch utility, 5-1, 5-22
DELETE parameter
CWAASDUT batch utility, 6-8
CWDDLPUT batch utility, 7-10
CWDDSUTL batch utility, 4-9
CWFXSDUT batch utility, 5-8
DELETEIC parameter
CWAASDUT batch utility, 6-8
CWFXSDUT batch utility, 5-8
deleting
Abend-AID report DDIO files, 3-45
source listing DDIO files, 3-19
DETACH command
CWAASDUT batch utility, 6-2, 6-22
CWDDLPUT batch utility, 7-2, 7-30
CWFXSDUT batch utility, 5-2, 5-24
determine average member size
DDIO files, 2-13
DIAGNOSE parameter, PROCESSOR command
Assembler Language Processor, 11-17
DIRDD parameter
CWAASDUT batch utility, 6-8
CWDDLPUT batch utility, 7-10
CWFXSDUT batch utility, 5-8
DIRDSN parameter
CWAASDUT batch utility, 6-8
CWDDLPUT batch utility, 7-10
CWFXSDUT batch utility, 5-8
DIRECTORY command
CWAASDUT batch utility, 6-2, 6-23
CWDDLPUT batch utility, 7-2, 7-31
CWDDSUTL batch utility, 4-1, 4-17
CWFXSDUT batch utility, 5-2, 5-25
directory entry, G-2
DIRENTS parameter
CWAASDUT batch utility, 6-8
CWDDLPUT batch utility, 7-10
CWFXSDUT batch utility, 5-9
DIRS command
CWAASDUT batch utility, 6-2, 6-24
CWDDLPUT batch utility, 7-2, 7-32
CWDDSUTL batch utility, 4-1, 4-18
CWFXSDUT batch utility, 5-2, 5-26
DIRVOL parameter
CWAASDUT batch utility, 6-9
CWDDLPUT batch utility, 7-10
CWFXSDUT batch utility, 5-9
DIRX command
CWAASDUT batch utility, 6-2, 6-23
CWDDLPUT batch utility, 7-2, 7-31
CWDDSUTL batch utility, 4-2, 4-17
CWFXSDUT batch utility, 5-2, 5-25
DIRX report, G-2
DIRX report for allocation DDIO files, 2-10
DIRX reports
Abend-AID report file DIRX report, 2-24
creating, 2-4
shared directory DIRX report, 2-36
Source Listing Database DIRX report, 2-49
source listing file DIRX report, 2-17
Source Listing Shared Directory (SRCEDIR) DIRX report, 2-56
transaction database DIRX report, 2-30
using the DIRX report, 2-17
using the previous DIRX report format, 2-17
displaying
Abend-AID report DDIO files, 3-48, 8-21
Abend-AID report file information, 3-46
source listing DDIO files, 3-22, 8-21
Distributed Viewing Support, 8-2
Compuware/VF Entry Panel, 8-2
documentation, related, xxiv
DSN parameter
CWAASDUT batch utility, 6-9
CWDDLPUT batch utility, 7-10
CWFXSDUT batch utility, 5-9
DUPS parameter
CWAASDUT batch utility, 6-9
CWDDLPUT batch utility, 7-11

CWDDSUTL batch utility, 4-9
CWFXSDUT batch utility, 5-9
DYNCREATE parameter
CWDDLPUT batch utility, 7-11

E

ENDDATE parameter
CWAASDUT batch utility, 6-9
CWDDLPUT batch utility, 7-11
CWDDSUTL batch utility, 4-9
CWFXSDUT batch utility, 5-9
ENDTIME parameter
CWAASDUT batch utility, 6-9
CWDDLPUT batch utility, 7-11
CWDDSUTL batch utility, 4-9
CWFXSDUT batch utility, 5-9
enhanced listing, G-2
enhanced listing, COBOL Language Processor, 9-10
entry, G-3
entry panel
browsing options, 8-6
Compuware/VF, 8-2
dataset choices, 8-3
processing options, 8-5
Viewing Facility, 8-3
environment, system, 1-2
ERRM File-overrides command
Assembler Language Processor, 11-12
C Language Processor, 12-11
COBOL Language Processor, 9-20
PL/I Language Processor, 10-14
error messages
CWDDUNLP return codes, C-2
mismatched source listing warnings, 8-31
Problem Documentation Utility, A-1, C-1
ERRORS parameter, PROCESSOR command
Assembler Language Processor, 11-18
C Language Processor, 12-16
COBOL Language Processor, 9-28
PL/I Language Processor, 10-20
errors, processing listings with
Assembler Language Processor, 11-5
C Language Processor, 12-5
COBOL Language Processor, 9-5
PL/I Language Processor, 10-5
ESTIMATE command
CWAASDUT batch utility, 6-2, 6-25
CWDDLPUT batch utility, 7-2, 7-33
CWFXSDUT batch utility, 5-2, 5-27
EXAMINE command
CWAASDUT batch utility, 6-2, 6-26
CWFXSDUT batch utility, 5-2, 5-28
EXEC parameter
CWDDSUTL batch utility, 4-9
executing
Abend-AID, 1-7
Abend-AID for CICS, 1-7
CSS, 1-7
XPEDITER/CICS, 1-7
XPEDITER/TSO, 1-7
execution pointer, effect of optimization logic
in C processing, 12-5
in COBOL processing, 9-6
EXPAND command
CWAASDUT batch utility, 6-2, 6-27
CWDDLPUT batch utility, 7-2, 7-34
CWFXSDUT batch utility, 5-2, 5-30
EXPORT command
CWAASDUT batch utility, 6-2, 6-28
CWDDLPUT batch utility, 7-2, 7-35
CWDDSUTL batch utility, 4-2, 4-19
CWFXSDUT batch utility, 5-2, 5-31
exporting
Abend-AID report file members, 3-50
source listing file members, 3-26
EXTCSS load library, B-2–B-3
EXTENTS parameter, 2-5
CWAASDUT batch utility, 6-9
CWDDLPUT batch utility, 7-11
CWFXSDUT batch utility, 5-9
EXTENTS=
example, 2-14

F

FDBDCUST DD statement
CWFXSDUT batch utility, 5-3
file access method
selecting, 2-6
File-AID, accessing from Compuware/VF, 8-27, 8-44
File-overrides command
Assembler Language Processor, 11-3
C Language Processor, 12-3
COBOL Language Processor, 9-3
PL/I Language Processor, 10-3
File-overrides commands, Assembler
DECK, 11-12
LOAD, 11-12
optional override values, 11-12
PRTI, 11-11
PRTO, 11-11
WRK0-WRK4, 11-11
File-overrides commands, C
ERRM, 12-11
LOAD, 12-12
optional override values, 12-12
PRTI, 12-11
PRTO, 12-11
File-overrides commands, COBOL
DECK, 9-20
ERRM, 9-20
LOAD, 9-20
optional override values, 9-20
PRTI, 9-19
PRTO, 9-19
WRK0-WRK4, 9-19
File-overrides commands, PL/I
ERRM, 10-14
LOAD, 10-14
optional override values, 10-14
PRTI, 10-13
PRTO, 10-14
WRK0-WRK4, 10-13
FIND command, Compuware/VF, 8-43
FLAG command, CWDDSUTL
BASEDDIO parameter, 4-8
BASENUMBER parameter, 4-8
CURRDDIO parameter, 4-8
CURRNUMBER parameter, 4-8
description, 4-2
FLAGGEDDDIO parameter, 4-10

repeated use, 4-20
report output, 4-21
supported compilers, 4-21
syntax, 4-22
FLAGGEDDDIO parameter
CWDDSUTL batch utility, 4-10
FORMAT command
CWDDSUTL batch utility, 4-2, 4-23
CWDDSUTL syntax, 4-24
formatting
Abend-AID report DDIO files, 3-38
DDIO file, G-3
DDIO files, 2-1, 2-14
source listing DDIO files, 3-12
formatting, CWDDLPUT
allocating and formatting new shared directory, 7-23
allocating and formatting new Source Listing Database, 7-24
pre-allocated shared directory, 7-23
pre-allocated Source Listing Database, 7-25
formatting, CWDDLPUT batch utility
pre-allocated Shared Directory, 7-23
FROMDD parameter
CWAASDUT batch utility, 6-9
CWDDLPUT batch utility, 7-11
CWDDSUTL batch utility, 4-10
CWFXSDUT batch utility, 5-10
FROMDSN parameter
CWAASDUT batch utility, 6-9
CWDDLPUT batch utility, 7-12
CWFXSDUT batch utility, 5-10
FROMOBJECT parameter
CWAASDUT batch utility, 6-9
CWDDLPUT batch utility, 7-12
CWFXSDUT batch utility, 5-10
FROMVOL parameter
CWAASDUT batch utility, 6-9
CWDDLPUT batch utility, 7-12
CWFXSDUT batch utility, 5-10
FrontLine support web site, xxiv
FS command, Compuware/VF, 8-43
full DDIO file, G-3
functions of Security Exit Program, 13-1

G

GROUPCOUNT parameter, 2-5
calculate value, 2-13
CWAASDUT batch utility, 6-10
CWDDLPUT batch utility, 7-12
CWDDSUTL batch utility, 4-10
CWFXSDUT batch utility, 5-10

H

Hardcopy Options screen, Compuware/VF, 8-19
HCI
description, 1-8
help, getting, xxvi
HFS files, use of, 12-5
Host Communications Interface, G-3
HTML documentation, xxiv

I

ID parameter
CWDDSUTL batch utility, 4-10
IMPORT command
CWAASDUT batch utility, 6-2, 6-30
CWDDLPUT batch utility, 7-2, 7-37
CWDDSUTL batch utility, 4-2, 4-25
CWFXSDUT batch utility, 5-1–5-2, 5-33
importing
Abend-AID report file members, 3-51
source listing file members, 3-27
INC parameter
CWAASDUT batch utility, 6-10
CWDDLPUT batch utility, 7-12
CWDDSUTL batch utility, 4-10
CWFXSDUT batch utility, 5-10
INCLUDE command
Assembler Language Processor, 11-3, 11-14
C Language Processor, 12-3, 12-13
COBOL Language Processor, 9-3, 9-22
PL/I Language Processor, 10-3, 10-16
INDD parameter
CWDDSUTL batch utility, 4-10
INDEX parameter, 2-5
CWDDSUTL batch utility, 4-11
INMEM configuration source member, B-2
inputs
Security Exit Program, 13-3
INSRC dataset name, B-2
Installation JCL Customization Facility, G-3
Internet, Compuware WWW address, xxv
invoking
Compuware/VF, 8-1
CSS Utilities, 3-2

J

JCL
creating for preprocessing and postprocessing, 3-4
JESID parameter
CWAASDUT batch utility, 6-10
CWDDSUTL batch utility, 4-11
JOBNAME parameter
CWDDSUTL batch utility, 4-11

L

LABEL parameter
CWAASDUT batch utility, 6-10
CWDDLPUT batch utility, 7-12
CWDDSUTL batch utility, 4-11
CWFXSDUT batch utility, 5-10
LANGPARM command
Assembler Language Processor, 11-3, 11-15
C Language Processor, 12-3, 12-14
COBOL Language Processor, 9-3, 9-23
PL/I Language Processor, 10-3, 10-17
LANGUAGE command
Assembler Language Processor, 11-3, 11-15
C Language Processor, 12-3, 12-14
COBOL Language Processor, 9-3, 9-24

PL/I Language Processor, 10-3, 10-17
LANGUAGE parameter
CWDDSUTL batch utility, 4-11
Language Processor (LP), G-3
Assembler Language Processor, 11-1
C Language Processor, 12-1
COBOL Language Processor, 9-1
functions, 3-1
PL/I Language Processor, 10-1
postprocessor, Assembler, 11-2
postprocessor, C, 12-2
postprocessor, COBOL, 9-2
postprocessor, PL/I, 10-2
preprocessor, 9-1
preprocessor, Assembler, 11-1
preprocessor, C, 12-1
preprocessor, PL/I, 10-1
run using your source program, A-2
supported languages, 1-4
LAU, G-3
license certificate, G-3
license file, G-3
License Management System, G-3
limitations
DDIO files, 2-12
line command, G-3
LINESIZE parameter
CWDDSUTL batch utility, 4-12
LMVERIFY, G-3
LOAD File-overrides command
Assembler Language Processor, 11-12
C Language Processor, 12-12
COBOL Language Processor, 9-20
PL/I Language Processor, 10-14
LOCATE command
Compuware/VF, 8-16
LOCK command
CWAASDUT batch utility, 6-2, 6-31
CWDDLPUT batch utility, 7-2, 7-38
CWDDSUTL batch utility, 4-2, 4-26
CWFXSDUT batch utility, 5-2, 5-34
locked member, G-3
locking DDIO members, 2-7
LPAR, G-3
licensing, G-3

M

manual lock, G-3
manuals, related, xxiv
messages
CWDDUNLP return codes, C-2
messages, preventing truncated, COBOL, 9-5
MGMTCLAS parameter
CWAASDUT batch utility, 6-10
CWDDLPUT batch utility, 7-12
CWFXSDUT batch utility, 5-11
MIPS, G-3
mismatched source listing warnings, 8-31
MODEL parameter
CWAASDUT batch utility, 6-10
CWDDLPUT batch utility, 7-13
CWFXSDUT batch utility, 5-11
MONTHSOLD parameter
CWAASDUT batch utility, 6-11
CWDDLPUT batch utility, 7-13
CWDDSUTL batch utility, 4-12
CWFXSDUT batch utility, 5-11
MOVE command
CWAASDUT batch utility, 6-2, 6-32
CWDDLPUT batch utility, 7-2, 7-39
CWDDSUTL batch utility, 4-2, 4-27
CWFXSDUT batch utility, 5-2, 5-35
moving
Abend-AID report file members, 3-50
source listing file members, 3-26

N

NEXT command
Compuware/VF, 8-17
NOTFOUND parameter
CWAASDUT batch utility, 6-11
CWDDLPUT batch utility, 7-13
CWDDSUTL batch utility, 4-12
CWFXSDUT batch utility, 5-11

O

OBJECT parameter
CWAASDUT batch utility, 6-11
CWDDLPUT batch utility, 7-13
CWFXSDUT batch utility, 5-11
OFFSET parameter, PROCESSOR command
COBOL Language Processor, 9-30
oldest member, G-3
OPTFILE compiler option, 12-5
optimization logic, effects of
in C processing, 12-5
in COBOL processing, 9-6
in PL/I processing, 10-5
options
for supported Assemblers, 11-6
for supported C compilers, 12-7
for supported COBOL compilers, 9-9
for supported PL/I compilers, 10-10
OUTDD parameter
CWDDLPUT batch utility, 7-13
CWDDSUTL batch utility, 4-12
OUTMEM member name, B-3
OUTPDS dataset name, B-3
output considerations, COBOL, 9-5
output information
Security Exit Program, 13-6
OUTPUT parameter, Assembler
DDIO command, 11-10
OUTPUT parameter, C
DDIO command, 12-10
OUTPUT parameter, COBOL
COBOL command, 9-14
DDIO command, 9-17
PRINT command, 9-26
PROCESSOR command, 9-29
special considerations, 9-6
OUTPUT parameter, PL/I
DDIO command, 10-12
PL/I command, 10-18
PRINT command, 10-19
PROCESSOR command, 10-20
Output Selection Menu (Compuware/VF)

Abend-AID, 8-21
outputs
Security Exit Program, 13-3

P

PAGESIZE parameter
CWDDSUTL batch utility, 4-12
parameters
AUTODELETE, 2-5
BLKSIZE, 2-5
EXTENTS, 2-5
GROUPCOUNT, 2-5
INDEX, 2-5
REPTCOUNT, 2-6
parameters, Assembler
$ASUP, 11-18
$NG, 11-18
$NIDR, 11-19
$NO, 11-19
BYPASS, 11-17
DIAGNOSE, 11-17
ERRORS, 11-18
parameters, C
$NIDR, 12-16
$NO, 12-16
ERRORS, 12-16
parameters, COBOL
$AOFF, 9-32
$ASUP, 9-32
$NG, 9-32
$NIDR, 9-33
$NO, 9-33
BYPASS, 9-28
ERRORS, 9-28
OFFSET, 9-30
OUTPUT, 9-29
POSITION, 9-29
SCALELINE, 9-30
STATISTICS, 9-30
TEXT, 9-31
WARNINGS, 9-31
parameters, CWAASDUT
AM, 6-5
AUTODELETE (AD), 6-5
BLKSIZE (BLK), 6-7
CISIZE, 6-7
CONTINUE, 6-7
CYLINDERS (CYL), 6-7
DATABASEDISP, 6-7
DATACLAS, 6-8
DAYSOLD, 6-8
DDNAME (DDN), 6-8
DELETE, 6-8
DELETEIC, 6-8
DIRDD, 6-8
DIRDSN, 6-8
DIRENTS, 6-8
DIRVOL, 6-9
DSN, 6-9
DUPS, 6-9
ENDDATE (EDATE), 6-9
ENDTIME (ETIME), 6-9
EXTENTS, 6-9
FROMDD, 6-9
FROMDSN, 6-9
FROMOBJECT, 6-9
FROMVOL, 6-9
GROUPCOUNT (GC), 6-10
INC, 6-10
JESID, 6-10
LABEL, 6-10
MGMTCLAS, 6-10
MODEL, 6-10
MONTHSOLD, 6-11
NOTFOUND, 6-11
OBJECT (OBJ), 6-11
REALLOCATE (REALLO), 6-11
RECORDS, 6-11
REPAIR, 6-11
REPTCOUNT (RC), 6-11
REPTNUMBER (RN), 6-11
RETAINRN, 6-12
SECONDARY, 6-12
SORT, 6-12
STARTDATE (SDATE), 6-12
STARTTIME (STIME), 6-12
STORCLAS, 6-13
TODD, 6-13
TODSN, 6-13
TOOBJECT, 6-13
TOVOL, 6-13
TRACKS (TRKS), 6-13
TYPE, 6-13
UNIT, 6-13
VOLSER (VOL), 6-13
VSAMCAT, 6-13
YEARSOLD, 6-13
parameters, CWDDLPUT
AM, 7-6
AUTODELETE, 7-6
BLKSIZE, 7-9
CISIZE, 7-9
CONTINUE, 7-9
CYLINDERS, 7-9
DATABASEDISP, 7-9
DATACLAS, 7-9
DAYSOLD, 7-9
DDNAME, 7-10
DELETE, 7-10
DIRDD, 7-10
DIRDSN, 7-10
DIRENTS, 7-10
DIRVOL, 7-10
DSN, 7-10
DUPS, 7-11
DYNCREATE, 7-11
ENDDATE, 7-11
ENDTIME, 7-11
EXTENTS, 7-11
FROMDD, 7-11
FROMDSN, 7-12
FROMOBJECT, 7-12
FROMVOL, 7-12
GROUPCOUNT, 7-12
INC, 7-12
LABEL, 7-12
MGMTCLAS, 7-12
MODEL, 7-13
MONTHSOLD, 7-13
NOTFOUND, 7-13
OBJECT, 7-13
OUTDD, 7-13
PROGRAM, 7-13

REALLOCATE, 7-14
RECORDS, 7-14
REPTCOUNT, 7-14
REPTNUMBER, 7-14
RETAINRN, 7-14
SECONDARY, 7-14
SORT, 7-14
STARTDATE, 7-15
STARTTIME, 7-15
STORCLAS, 7-15
TODD, 7-15
TODSN, 7-15
TOOBJECT, 7-15
TOVOL, 7-15
TRACKS, 7-16
TYPE, 7-16
UNIT, 7-16
VOLSER, 7-16
VSAMCAT, 7-16
YEARSOLD, 7-16
parameters, CWDDSUTL
ABEND, 4-6
AUTODELETE, 4-6
BASEDDIO (BDDIO), 4-8
BASENUMBER (BRN), 4-8
BLKSIZE (BLK), 4-8
CONTINUE, 4-8
CURDDIO (CDDIO), 4-8
CURRNUMBER (CRN), 4-8
DAYSOLD, 4-9
DELETE, 4-9
DUPS, 4-9
ENDDATE (EDATE), 4-9
ENDTIME (ETIME), 4-9
EXEC, 4-9
EXTENTS, 4-10
FLAGGEDDDIO (FDDIO), 4-10
FROMDD, 4-10
GROUPCOUNT (GC), 4-10
ID, 4-10
INC, 4-10
INDD, 4-10
INDEX, 4-11
JESID, 4-11
JOBNAME, 4-11
LABEL, 4-11
LANGUAGE (LANG), 4-11
LINESIZE (LIN), 4-12
MONTHSOLD, 4-12
NOTFOUND, 4-12
OUTDD (O), 4-12
PAGESIZE (PG), 4-12
PROGRAM (P), 4-12
REPTCOUNT (RC), 4-12
REPTNUMBER (RN), 4-12
SECONDARY, 4-12
SEEFLAGS, 4-13
SLSDD, 4-13
SORT, 4-13
STARTDATE (SDATE), 4-13
STARTTIME (STIME), 4-14
SUBTYPE, 4-14
TODD, 4-14
TYPE, 4-14
YEARSOLD, 4-14
parameters, CWFXSDUT
AM, 5-5
APPLID, 5-6
AUTODELETE (AD), 5-6
BLKSIZE (BLK), 5-7
CICSCODE (CC), 5-7
CISIZE, 5-7
CONTINUE, 5-7
CYLINDERS (CYL), 5-8
DATABASEDISP, 5-8
DATACLAS, 5-8
DAYSOLD, 5-8
DDNAME (DDN), 5-8
DELETE, 5-8
DELETEIC, 5-8
DIRDD, 5-8
DIRDSN, 5-8
DIRENTS, 5-9
DIRVOL, 5-9
DSN, 5-9
DUPS, 5-9
ENDDATE (EDATE), 5-9
ENDTIME (ETIME), 5-9
EXTENTS, 5-9
FROMDD, 5-10
FROMDSN, 5-10
FROMOBJECT, 5-10
FROMVOL, 5-10
GROUPCOUNT (GC), 5-10
INC, 5-10
LABEL, 5-10
MGMTCLAS, 5-11
MODEL, 5-11
MONTHSOLD, 5-11
NOTFOUND, 5-11
OBJECT (OBJ), 5-11
REALLOCATE (REALLO), 5-11
REPAIR, 5-12
REPTCOUNT (RC), 5-12
REPTNUMBER (RN), 5-12
RETAINRN, 5-12
SECONDARY, 5-12
SORT, 5-12
STARTDATE (SDATE), 5-13
STARTTIME (STIME), 5-13
STORCLAS, 5-13
TODD, 5-13
TODSN, 5-13
TOOBJECT, 5-13
TOVOL, 5-13
TRACKS (TRKS), 5-13
TRAN (T), 5-13
TYPE, 5-14
UNIT, 5-14
VOLSER (VOL), 5-14
VSAMCAT, 5-14
YEARSOLD, 5-14
parameters, PL/I
$NG, 10-21
$NO, 10-21
BYPASS, 10-20
ERRORS, 10-20
OUTPUT, 10-20
PDF documentation, xxiv
PL/I Language Processor, G-4
%NOPRINT, 1-5
compilers, 1-2
CONDDDIO command, 10-3, 10-11
CONDSTEP command, 10-3, 10-11
DDIO command, 10-3, 10-12
DECK File-overrides command, 10-14

description of, 10-1
effect of optimization logic on execution pointer, 10-5
ERRM File-overrides command, 10-14
File-overrides commands, 10-3, 10-13
File-overrides, optional override values, 10-14
handling suppressed statements, 10-5
INCLUDE command, 10-3, 10-16
insufficient primary space allocation, 10-5
LANGPARM command, 10-3, 10-17
LANGUAGE command, 10-3, 10-17
LOAD File-overrides command, 10-14
PL/I command, 10-3, 10-18
PL/I Parameter, LANGUAGE command, 10-17
PRINT command, 10-3, 10-19, 11-6
processing listings with errors, 10-5
PROCESSOR command, 10-3, 10-20
PROGRAM-NAME, 10-22
PROGRAM-NAME command, 10-3
PRTI File-overrides command, 10-13
PRTO File-overrides command, 10-14
required compiler options, 10-10
restrictions, 10-6
sample JCL and program names, 10-4
SYSTEM command, 10-3, 10-22
usage considerations, 10-5
viewtime restriction, 8-21
WRK0-WRK4 File-overrides commands, 10-13
pointer affected by optimization logic
in PL/I processing, 10-5
POSITION parameter, PROCESSOR command
COBOL Language Processor, 9-29
postprocessor, Assembler
benefits, 11-2
description of, 11-2
File-overrides commands, 11-3, 11-11
postprocessor, C
benefits, 12-2
description of, 12-2
File-overrides commands, 12-3, 12-11
postprocessor, COBOL
benefits, 9-2
CSPPRTO file, 9-5
description of, 9-2
File-overrides commands, 9-19
WORD option, 9-5
postprocessor, CSS Utilities, 3-7
postprocessor screen fields, 3-8
running the postprocessor, 3-8
screen fields, 3-8
selecting members for processing, 3-10
submitting JCL for processing, 3-10
using sample JCL, 3-8
postprocessor, PL/I
benefits, 10-2
description of, 10-2
File-overrides commands, 10-3, 10-13
preprocessor
automated compiler options, 1-6
benefits, 1-5
capturing suppressed source code, 1-5
handling compile errors, 1-5
simplified JCL, 1-6
steps, 1-5
preprocessor, Assembler
automated compiler options, 11-2
benefits, 11-1
capturing suppressed source code, 11-1
CONDDDIO command, 11-3, 11-8
CONDSTEP command, 11-3, 11-8
description of, 11-1
dynamically allocated files, 11-2
handling compiler errors, 11-1
INCLUDE command, 11-3, 11-14
LANGPARM command, 11-3, 11-15
simplified JCL, 11-2
preprocessor, C
automated compiler options, 12-2
benefits, 12-1
CONDDDIO command, 12-3, 12-9
CONDSTEP command, 12-3, 12-9
description of, 12-1
dynamically allocated files, 12-2
handling compiler errors, 12-1
LANGPARM command, 12-3, 12-14
simplified JCL, 12-2
preprocessor, COBOL
benefits, 9-1
CONDDDIO command, 9-3, 9-15
CONDERRL command, 9-3, 9-15
CONDSTEP command, 9-3, 9-16
CWPPRTIDD statement, 9-6
description of, 9-1
dynamically allocated files, 9-2
INCLUDE command, 9-3, 9-22
LANGPARM command, 9-3, 9-23
suppressed statements, 9-6
SYSPRINT file, 9-5
preprocessor, CSS Utilities, 3-5
selecting members for processing, 3-7
submitting JCL for processing, 3-7
preprocessor, PL/I
automated compiler options, 10-2
benefits, 10-1
capturing suppressed source code, 10-2
CONDSTEP command, 10-3, 10-11
description of, 10-1
dynamically allocated files, 10-2
INCLUDE command, 10-3, 10-16
LANGPARM command, 10-3, 10-17
simplified JCL, 10-2
primary command, G-4
primary space allocation
Assembler Language Processor, 11-5
C Language Processor, 12-5
COBOL Language Processor, 9-6
PL/I Language Processor, 10-5
PRINT command
Assembler Language Processor, 11-3, 11-16
C Language Processor, 12-3, 12-15
COBOL Language Processor, 9-3, 9-26
CWDDLPUT batch utility, 7-2, 7-40
CWDDSUTL batch utility, 4-2, 4-28
PL/I Language Processor, 10-3
printing
Hardcopy Options screen, 8-19
Problem Documentation Utility, G-4
Problem Documentation Utility, using, A-1, C-1
PROCESSOR command
Assembler Language Processor, 11-3, 11-17
C Language Processor, 12-3, 12-16
COBOL Language Processor, 9-3, 9-28
PL/I Language Processor, 10-3, 10-20
program listing section, 8-25
PROGRAM parameter
CWDDLPUT batch utility, 7-13

CWDDSUTL batch utility, 4-12
program storage
selecting, 8-23
viewing, 8-24
program storage section, 8-23
PROGRAM-NAME command
Assembler Language Processor, 11-3, 11-19
C Language Processor, 12-3, 12-17
COBOL Language Processor, 9-3, 9-34
PL/I Language Processor, 10-3, 10-22
PROGRAM-NAME parameter
DDIO command, 11-9
PRTI File-overrides command
Assembler Language Processor, 11-11
C Language Processor, 12-11
COBOL Language Processor, 9-19
PL/I Language Processor, 10-13
PRTO File-overrides command
Assembler Language Processor, 11-11
C Language Processor, 12-11
COBOL Language Processor, 9-19
PL/I Language Processor, 10-14

R

REALLOCATE parameter
CWAASDUT batch utility, 6-11
CWDDLPUT batch utility, 7-14
CWFXSDUT batch utility, 5-11
reconstructing a configuration source member, B-3
RECORDS parameter
CWAASDUT batch utility, 6-11
CWDDLPUT batch utility, 7-14
CWFXSDUT batch utility, 5-11
reformatting
Abend-AID report DDIO files, 3-38
source listing DDIO files, 3-12
register entries
Security Exit Program, 13-3
related publications, xxiv
renaming report files
CWAASDUT, 6-3
renaming shared directories
CWAASDUT, 6-3
CWDDLPUT, 7-3
CWFXSDUT, 5-3
renaming transaction databases
CWDDLPUT, 7-3
CWFXSDUT, 5-3
REORDER command
CWDDSUTL batch utility, 4-2, 4-29
REPAIR parameter
CWAASDUT batch utility, 6-11
CWFXSDUT batch utility, 5-12
report file, G-4
renaming, CWAASDUT batch utility, 6-3
reports
Abend-AID report file DIRX report, 2-24
shared directory DIRX report, 2-36
Source Listing Database DIRX report, 2-49
source listing file DIRX report, 2-17
Source Listing Shared Directory (SRCEDIR) DIRX report, 2-56
transaction database DIRX report, 2-30
using the DIRX report, 2-17
using the previous DIRX report format, 2-17
REPTCOUNT parameter, 2-6
CWAASDUT batch utility, 6-11
CWDDLPUT batch utility, 7-14
CWDDSUTL batch utility, 4-12
CWFXSDUT batch utility, 5-12
REPTNUMBER parameter
CWAASDUT batch utility, 6-11
CWDDLPUT batch utility, 7-14
CWDDSUTL batch utility, 4-12
CWFXSDUT batch utility, 5-12
required options
for supported Assemblers, 11-6
for supported C compilers, 12-7
for supported COBOL compilers, 9-9
for supported PL/I compilers, 10-10
reserved word tables, COBOL, 9-5
RESET command
CWAASDUT batch utility, 6-2, 6-33
CWDDLPUT batch utility, 7-2, 7-41
CWFXSDUT batch utility, 5-2, 5-36
restricted zaps
retrofitting, B-1
restrictions
Assembler Language Processor, 11-6
C Language Processor, 12-6
COBOL Language Processor, 9-7
CWAASDUT, 6-4
CWDDLPUT batch utility, 7-5
CWDDSUTL, 4-4
CWFXSDUT, 5-4
PL/I Language Processor, 10-6
RETAINRN parameter
CWAASDUT batch utility, 6-12
CWDDLPUT batch utility, 7-14
CWFXSDUT batch utility, 5-12
return codes
CWDDUNLP source extraction utility, C-2
revisions, xix
running the postprocessor
CSS Utilities, 3-8
running the preprocessor
CSS Utilities, 3-5

S

sample JCL and programs
Assembler Language Processor, 11-4
C Language Processor, 12-4
COBOL Language Processor, 9-4
PL/I Language Processor, 10-4
SCALELINE parameter
COBOL Language Processor, 9-30
SCRATCH command
CWAASDUT batch utility, 6-3, 6-34
CWDDLPUT batch utility, 7-2, 7-41
CWFXSDUT batch utility, 5-2, 5-36
SECONDARY parameter
CWAASDUT batch utility, 6-12
CWDDLPUT batch utility, 7-14
CWDDSUTL batch utility, 4-12
CWFXSDUT batch utility, 5-12
SECUCMND field, 13-3
SECUCUSR field, 13-3
Security Exit Program, 13-1, G-4
Abend-AID for CICS report files, 13-2
Abend-AID report files, 13-2

CSS components, 13-1
functions of, 13-1
inputs and outputs, 13-3
output information, 13-6
register entries, 13-3
source listing files, 13-2
SEEFLAGS parameter
CWDDSUTL batch utility, 4-13
selecting
Abend-AID report file members, 3-52
members for processing, 3-10
server grace, G-4
setting Abend-AID report file options, 3-37
setting advanced create/format parameters, 3-15, 3-42
setting advanced parameters, 3-29
setting source listing file options, 3-11
shared directories, 2-3
CWAASDUT, renaming, 6-3
CWDDLPUT, renaming, 7-3
CWFXSDUT, renaming, 5-3
description of, 2-3
shared directory DIRX report, 2-36
shared directory, G-4
sharing files
source listing files, 2-7
simplified JCL, 1-6
SLSDD parameter
CWDDSUTL batch utility, 4-13
SLSF001 DD statement
CWDDSUTL batch utility, 4-3
SMS parameters
DATACLAS, 5-8, 6-8
MGMTCLAS, 5-11, 6-10
STORCLAS, 5-13, 6-13
softcopy documentation, xxiv
SORT command
Compuware/VF, 8-17
sort keys
Abend-AID for CICS transaction database, 5-12
Abend-AID report file, 6-12
source listing file, 4-13, 8-18
SORT parameter
CWAASDUT batch utility, 6-12
CWDDLPUT batch utility, 7-14
CWDDSUTL batch utility, 4-13
CWFXSDUT batch utility, 5-12
SOURCE command
Compuware/VF, 8-41
source extraction utility
CWDDUNLP, 2-4
source extraction utility, CWDDUNLP
return codes, C-2
source listing, G-4
Source Listing Database, 2-3
creating, CWDDLPUT batch utility, 7-3
DIRX report, 2-49
source listing database, G-4
source listing DDIO files, 2-2, 3-1
advanced parameters, 3-29
copying members, 3-24
creating, 3-12
deleting, 3-19
displaying, 3-22, 8-21
displaying file information, 3-20
exporting members, 3-26
formatting, 3-12
functions, 3-1
importing members, 3-27
moving members, 3-26
selecting specific members for processing, 3-28
sharing, 2-7
sort keys, 4-13, 8-18
Source Listing File Facility screen, 3-11
Source Listing File Information screen, 3-20
working with, 3-10
working with file members, 3-21
source listing file, G-4
source listing files
DIRX report, 2-17
Security Exit Program, 13-2
source listing member
reducing the size of, 2-16
Source Listing Shared Directory (SRCEDIR) DIRX report, 2-56
SOURCE option
Compuware/VF, 8-7
CSS Utilities, 3-10
space allocation
Assembler Language Processor, 11-5
C Language Processor, 12-5
COBOL Language Processor, 9-6
PL/I Language Processor, 10-5
STARTDATE parameter
CWAASDUT batch utility, 6-12
CWDDLPUT batch utility, 7-15
CWDDSUTL batch utility, 4-13
CWFXSDUT batch utility, 5-13
STARTTIME parameter
CWAASDUT batch utility, 6-12
CWDDLPUT batch utility, 7-15
CWDDSUTL batch utility, 4-14
CWFXSDUT batch utility, 5-13
STATISTICS parameter, PROCESSOR command
COBOL Language Processor, 9-30
STORCLAS parameter
CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-15
CWFXSDUT batch utility, 5-13
STROBE, G-4
submitting JCL for processing, 3-7, 3-10
SUBTYPE parameter
CWDDSUTL batch utility, 4-14
support, obtaining, xxvi
suppressed statements, G-4
Assembler Language Processor, 11-5
COBOL Language Processor, 9-6
PL/I Language Processor, 10-5
preprocessor, COBOL, 9-6
synonyms
COBOL Language Processor, 9-5
syntax conventions, xxv
syntax diagrams, xxv
SYSPRINT
COBOL Language Processor, 9-5
SYSTEM command
Assembler Language Processor, 11-3, 11-20
C Language Processor, 12-3, 12-17
COBOL Language Processor, 9-3, 9-34
PL/I Language Processor, 10-3, 10-22
system environment, 1-2

T

technical support, xxvi

Problem Documentation Utility, A-1, C-1
TEST command
CWDDLPUT batch utility, 7-2, 7-42
TEXT parameter
COBOL Language Processor, 9-31
TIME parameter
DDIO command, 11-9
TODD parameter
CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-15
CWDDSUTL batch utility, 4-14
CWFXSDUT batch utility, 5-13
TODSN parameter
CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-15
CWFXSDUT batch utility, 5-13
TOOBJECT parameter
CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-15
CWFXSDUT batch utility, 5-13
TOVOL parameter
CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-15
CWFXSDUT batch utility, 5-13
TP subsystem, 1-8
TRACKS parameter
CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-16
CWFXSDUT batch utility, 5-13
TRAN parameter
CWFXSDUT batch utility, 5-13
transaction database, G-4
description of, 2-3
renaming, CWDDLPUT batch utility, 7-3
renaming, CWFXSDUT batch utility, 5-3
transaction database DIRX report, 2-30
transaction report, G-4
truncated COBOL messages, preventing, 9-5
tuning DDIO files, 2-15
TYPE parameter
CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-16
CWDDSUTL batch utility, 4-14
CWFXSDUT batch utility, 5-14

U

UNCONVERT command
CWDDLPUT batch utility, 7-3, 7-43
UNIT parameter
CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-16
CWFXSDUT batch utility, 5-14
UNIT temporary dataset, B-2
UNLOCK command
CWAASDUT batch utility, 6-3, 6-35
CWDDLPUT batch utility, 7-3, 7-44
CWDDSUTL batch utility, 4-2, 4-30
CWFXSDUT batch utility, 5-2, 5-37
unlocked member, G-4
update extracted member, B-3
usage considerations
Assembler Language Processor, 11-5
C Language Processor, 12-5
COBOL Language Processor, 9-4–9-5
CWDDSUTL, 4-2
PL/I Language Processor, 10-5
usage restrictions
Assembler Language Processor, 11-6
C Language Processor, 12-6
COBOL Language Processor, 9-7
CWAASDUT, 6-4
CWDDLPUT, 7-5
CWDDSUTL, 4-4
CWFXSDUT, 5-4
PL/I Language Processor, 10-6
USECSS load library to STEPLIB, B-2–B-3
using the DIRX report, 2-17
using the previous DIRX report format, 2-17
utilities, batch file
description of, 2-3
utilities, CSS
AAREPORT option, 3-37
Abend-AID report files, working with, 3-37
accessing from Compuware/VF, 8-44
CREATEJCL option, 3-4
functions of, 3-1
global settings, 3-3
invoking, 3-2
preparing compiler listings, 3-4
source listing files, working with, 3-10
SOURCE option, 3-10

V

verification report (LMVERIFY), G-4
VERIFY command
CWAASDUT batch utility, 6-3, 6-36
CWDDLPUT batch utility, 7-3, 7-45
CWFXSDUT batch utility, 5-2, 5-38
VIEW command, Compuware/VF, 8-42
VIEW option
Compuware/VF, 8-2
view time, G-4
Viewing Facility
AAREPORT option, 8-8
Abend-AID primary menu, 8-1
Abend-AID report files, viewing, 8-21
accessing CSS Utilities, 8-44
contact information, 8-9
creating JCL for language processing, 8-7
data locator, 8-28
dataset directory screen, 8-10
entry panel, 8-2
entry panel fields, 8-3
LOCATE command, 8-16
NEXT command, 8-17
Output Selection Menu, 8-21
preprocessing and postprocessing, 8-7
primary commands, 8-16
required parameters, 8-7
SORT command, 8-17
source listing files, viewing, 8-21
SOURCE option, 8-7
VIEW option, 8-2
VOLSER parameter
CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-16
CWFXSDUT batch utility, 5-14
VSAM space
allocated per track by CISIZE, 2-12
VSAMCAT parameter

CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-16
CWFXSDUT batch utility, 5-14

W

WARNINGS parameter command
COBOL Language Processor, 9-31
WORD option, 9-5
working with Source Listing File Members, 3-21
World Wide Web, Compuware address, xxv
WRK0-WRK4 File-overrides commands
Assembler Language Processor, 11-11
COBOL Language Processor, 9-19
PL/I Language Processor, 10-13

X

XPEDITER/CICS
executing, 1-7
XPEDITER/TSO
executing, 1-7

Y

YEARSOLD parameter
CWAASDUT batch utility, 6-13
CWDDLPUT batch utility, 7-16
CWDDSUTL batch utility, 4-14
CWFXSDUT batch utility, 5-14