

z/VM
7.4

DFSMS/VM Storage Administration



Note:

Before you use this information and the product it supports, read the information in [“Notices” on page 265](#).

This edition applies to version 7, release 4 of IBM® z/VM® (product number 5741-A09) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2024-09-18

© **Copyright International Business Machines Corporation 1991, 2024.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures.....	xi
Tables.....	xvii
About This Document.....	xix
Intended Audience.....	xix
Syntax, Message, and Response Conventions.....	xix
Where to Find More Information.....	xxii
Links to Other Documents and Websites.....	xxii
How to provide feedback to IBM.....	xxiii
Summary of Changes for z/VM: DFSMS/VM Storage Administration	xxv
SC24-6279-74, z/VM 7.4 (September 2024).....	xxv
SC24-6279-73, z/VM 7.3 (December 2023).....	xxv
SC24-6279-73, z/VM 7.3 (September 2022).....	xxv
SC24-6279-01, for z/VM 7.2 (September 2020).....	xxv
Part 1. System Managed Storage and DFSMS/VM.....	1
Chapter 1. System Managed Storage Concepts.....	3
What is System Managed Storage?.....	3
Goal of System Managed Storage.....	3
Space Management Considerations for the Storage Administrator.....	3
Availability Management.....	4
Performance Management.....	4
SMS Constructs.....	5
Storage Administration Groups.....	5
The Storage Administrator.....	5
Storage Management in z/VM.....	5
What is Secondary Storage?.....	6
Chapter 2. DFSMS/VM Function Level 221.....	7
Minidisk Management.....	7
Space Management.....	7
Management Class.....	7
Rules and Usage of Management Classes.....	7
ACS Processing.....	8
Configurations.....	9
DFSMS/VM Control File.....	11
The DFSMS/VM Interface.....	11
DFSMS/VM and the Shared File System.....	11
Shared File System References.....	15
Part 2. Storage Administration Guide.....	17
Chapter 3. Using the Interactive Storage Management Facility.....	19
Entering and Exiting ISMF.....	19
Entering ISMF.....	19

Exiting ISMF.....	19
Understanding the Relationship of ISMF to DFSMS.....	20
Working in the ISMF Environment.....	20
Types of ISMF Panels.....	20
Identifying an ISMF Panel.....	21
Initiating Action from an ISMF Panel.....	21
Moving Through ISMF Panels.....	21
Moving Around ISMF List Panels.....	22
Selecting an ISMF Task.....	23
Changing the User Profile.....	23
Profile Application.....	24
File Application.....	25
Minidisk Application.....	27
Management Class Application.....	28
Automatic Class Selection (ACS) Application.....	29
Configuration Application.....	29
List Application.....	30
Getting Online Help While Using ISMF.....	32
Help Messages.....	32
Task-Related Help Panels.....	32
Help Index.....	33
Moving Through Help Panels.....	33
Displaying Program Function Key Settings.....	33
Setting Program Function (PF) Keys.....	34
 Chapter 4. Management Classes.....	 39
Contents of a Management Class.....	39
Management Class Name.....	39
Description.....	39
Expiration Attributes.....	39
Migration Attributes.....	40
Summary of the Attributes.....	41
Assigning Management Classes.....	42
Working with ISMF for Management Class Application Tasks.....	43
Management Class Application Selection Panel.....	43
 Chapter 5. Configurations.....	 53
Configuration Application.....	53
Configuration Application Selection Panel.....	54
Configuration Base Display.....	55
Base Configuration Define or Alter.....	56
Validating the Source Configuration.....	58
Validation Listing.....	59
Activating a Configuration.....	59
 Chapter 6. Automatic Class Selection.....	 63
ACS Processing.....	63
Determining When to Invoke ACS Processing.....	63
Customizing ACS Processing.....	64
ACS Processing Steps.....	65
Automatic Class Selection Application.....	66
Using ISMF for Automatic Class Selection Tasks.....	66
Edit ACS Routines.....	67
Translate ACS Routines.....	67
ACS Test Selection Panel.....	68
ACS Test Case Define or ACS Test Case Alter.....	69
Test ACS Routines.....	71
ACS Test Listing.....	71

Display ACS Object Information.....	72
Deleting an ACS Object from a Source Configuration File.....	73
Output Listing Disposition.....	73
Chapter 7. Specifying DFSMS/VM Secondary Storage.....	75
Naming a Migration Level 1 File Space.....	75
Specifying No Secondary Storage.....	75
Determining the Amount of Migration Level 1 Space.....	76
Renaming Migration Level 1 and Subdirectory Names.....	76
Naming the Migration Level 1 Subdirectories.....	76
Renaming the Migration Level 1 File Pool.....	77
Renaming a Primary File Pool Containing Migrated Data.....	77
Renaming the Fully Qualified LU Name or Global Resource ID.....	78
Renaming the ML1 File Space.....	78
Abnormal Conditions.....	78
Chapter 8. DFSMS/VM Backup and Recovery Processing.....	81
Backup Processing.....	81
Using the SFS FILEPOOL BACKUP Utility.....	81
Using Non-IBM Backup Facilities.....	82
Recommendations.....	83
Recovery Processing.....	84
Recovering an Entire Storage Group.....	84
Recovering Specific Files.....	85
Recovering From a Secondary Storage Failure.....	85
Chapter 9. Working with a Minidisk List.....	87
Using ISMF for Minidisk Tasks.....	87
Creating a New Minidisk List.....	87
Saving a Minidisk List.....	91
Accessing a Saved Minidisk List.....	92
Chapter 10. Moving and Checking CMS Minidisks.....	93
Planning Move Activities.....	93
Customization Considerations.....	93
Grouping Minidisks for Move Requests.....	93
Selecting Move Options.....	94
Issuing Move Requests.....	95
Tracking Move Requests.....	95
Implementing the MOVE Process.....	96
Move Options.....	96
Move Process Results.....	98
Move Errors.....	98
Using the MOVE Line Operator.....	99
Using the MOVE List Command.....	102
Verifying the Integrity of CMS Minidisks.....	106
Check Options.....	106
Check Results.....	106
Issuing Check Requests.....	107
Request IDs.....	110
Chapter 11. Starting and Stopping DFSMS/VM Processing.....	111
Starting DFSMS/VM.....	111
Stopping the DFSMS Master and Server Machines.....	111
STOP QUIESCE.....	112
STOP IMMEDIATE.....	112
Chapter 12. User Authorization and Security Controls.....	113

Security Controls.....	113
Using RACF/VM for Security.....	113
DFSMS/VM Authorization File.....	113
Using a Different Facility for Security.....	114
Using an Installation-Wide Exit.....	114
The DFSMS Command Authorization Process.....	114
Security of File Spaces That DFSMS/VM Uses.....	115
Other Security Considerations.....	116
 Chapter 13. Managing Storage Using DFSMS/VM.....	117
Managing a Storage Group.....	117
Exit before Migration or Expiration of a File.....	120
The MIGRATE Command.....	120
Recall.....	121
Automatic Recall.....	121
Recall by Command.....	122
Converting a Storage Group.....	123
Reporting on DFSMS/VM Migrated Data.....	123
Example of Reporting on a File Space.....	123
Example of Reporting on a Storage Group.....	125
The DELETE Command.....	127
Deleting ML1 and ML2 Entries.....	127
Deleting ML2 Backup Entries.....	128
 Part 3. Reference.....	129
 Chapter 14. DFSMS Commands.....	131
Command Processing Overview.....	131
Commands Processed in the User Machine.....	131
Commands that Produce Reports.....	131
Operational Commands.....	132
Return Codes.....	132
Online HELP for Messages.....	133
Online HELP for DFSMS Commands.....	133
Pattern Matching.....	134
Using DFSMS Commands.....	134
DFSMS ACTIVATE.....	135
DFSMS ALTER.....	136
DFSMS CHECK.....	138
DFSMS CONVERT.....	140
DFSMS COPY.....	141
DFSMS DELETE.....	144
DFSMS DISCARD.....	146
DFSMS MANAGE.....	147
DFSMS MIGRATE.....	150
DFSMS MOVE.....	151
DFSMS QUERY.....	157
DFSMS RECALL.....	158
DFSMS REPORT.....	160
DFSMS STOP.....	161
Using ISMF Commands and Line Operators.....	162
ISMF Commands.....	162
ISMF Line Operators.....	165
 Chapter 15. User Extensions Reference.....	171
Initiating a User Extension.....	171
As a Command.....	171

As a Line Operator.....	172
Accessing a Saved List through EXECs.....	173
Initiating an EXEC as a Line Operator.....	173
Initiating an EXEC Outside of ISMF.....	174
Chapter 16. ACS Language Reference.....	177
ACS Language Constants.....	177
Simple Mask Rules.....	177
Directory Path Mask Rules.....	178
ACS Language Read/Write Variables.....	178
ACS Language Read-Only Variables.....	179
Comparison Operators.....	182
Comparison Rules.....	183
Boolean Expressions.....	183
ACS Directory-Level Indexing Function.....	183
ACS Language Statements.....	184
PROC.....	184
FILTLIST.....	185
SET.....	186
DO.....	187
IF.....	187
SELECT.....	188
EXIT.....	189
WRITE.....	189
END.....	190
Appendix A. ACS REXX Exit.....	191
General Information.....	191
Using the IBM Compiler for REXX/370.....	191
Using Interpreted REXX.....	192
Activation of the REXX Exit.....	192
Coding the REXX Exit.....	192
ACS Variables.....	192
DFSMS Subcommand Environment.....	195
DFSMS Subcommands.....	195
Returning Control to DFSMS/VM.....	197
Handling of Errors.....	198
Performance.....	198
Sample ACS REXX Routine for a Large System.....	199
Appendix B. ACS Module Exit.....	203
General Information.....	203
Attributes of the Module Exit.....	203
Activation of the Module Exit.....	204
Coding the Module Exit.....	204
Entry Parameters.....	204
Assigning a Management Class.....	205
Returning Messages.....	206
Returning Control to DFSMS/VM.....	206
Management Class Variables.....	207
Parameter List for the ACS Module Exit (IGDACSPM).....	207
Invoking the ACS Routine.....	213
ACS Routine Return Codes.....	215
Appendix C. Samples of DFSMS/VM Systems Applications.....	217
The ACS Routine.....	219
Converting SFS Storage Groups.....	220

Conversion in the Large System.....	220
Conversion in the Small System.....	221
Managing a Storage Group.....	221
Managing the Large Environment.....	221
Managing the Small System.....	223
Appendix D. Sample ACS Routines.....	225
Sample ACS Routine Without Migration to ML1.....	225
Sample ACS Routine With Migration to ML1.....	227
Sample ACS Routine With Migration to ML1 and ML2.....	229
Appendix E. Sample EXEC for Automated Migration.....	231
Appendix F. ISMF Variables.....	233
Control Variables for ISMF lists.....	233
File List Panel Column Variables.....	235
Minidisk List Panel Column Variables.....	236
Management Class List Panel Column Variables.....	237
Saved ISMF Lists Panel Column Variables.....	238
Column Variables for Saved Lists.....	239
Appendix G. Sample User Extension Program — PRNTMINI.....	241
PRNTMINI EXEC.....	241
SAMPLE FILE TAILORING SKELETON.....	245
Sample PRNTMINI Output.....	247
Appendix H. Implementing Configurations.....	249
How do I use a configuration to automatically assign management classes?.....	249
What are the contents of the configuration?.....	249
How can I make a new configuration?.....	250
How can I modify configurations?.....	250
What can I use to automatically assign management classes?.....	251
How do I put the exits, ACS routine, and configuration together and use them?.....	251
How do I stop using an ACS routine?.....	251
How do I stop using a module or REXX exit?.....	251
How do I use configurations and ACS to manage files at a directory level?.....	252
How do I assign the NULL management class?.....	252
How do I change management classes?.....	253
Appendix I. Multi-Language Support.....	255
Appendix J. DFSMS/VM Accounting Services.....	257
USER Accounting Record.....	257
User Accounting Record Command Codes.....	258
System Accounting Record.....	259
Appendix K. Year 2000 Enhancements for DFSMS/VM.....	261
Fixed Window Date Usage.....	261
Saved ISMF Lists.....	261
DFSMS COPY and MOVE commands.....	262
DFSMS/VM Generated Reports.....	262
DFSMS CHECK Command Report.....	262
DFSMS REPORT Command Report.....	263
Removal of ISMF Alphabetic Restriction on Directory Names.....	264
Notices.....	265

Programming Interface Information.....	266
Trademarks.....	266
Terms and Conditions for Product Documentation.....	266
IBM Online Privacy Statement.....	267
Bibliography.....	269
Where to Get z/VM Information.....	269
z/VM Base Library.....	269
z/VM Facilities and Features.....	270
Prerequisite Products.....	272
Related Products.....	272
Index.....	273

Figures

1. Example Environment with Configuration Use.....	10
2. Shared File System.....	12
3. SFS File Pool Structure.....	13
4. SFS Directory Structure.....	14
5. Primary Option Menu for General Users (with Copyright Window).....	19
6. ISMF Profile Option Menu.....	23
7. Primary Option Menu for Storage Administrators.....	24
8. File Application Selection Entry Panel (Part 1 of 3).....	25
9. File Application Selection Entry Panel (Part 2 of 3).....	25
10. File Application Selection Entry Panel (Part 3 of 3).....	26
11. File List Panel.....	26
12. Management Class Application Selection Panel for Storage Administrators.....	28
13. Configuration Application Selection Panel.....	30
14. Sample Saved ISMF Lists Panel.....	31
15. Sample Short Message.....	32
16. Sample Long Message.....	32
17. PF Key Defaults.....	35
18. PF Key Definitions and Labels—Primary Keys Panel.....	36
19. Management Class Application Selection Panel for Storage Administrator.....	43
20. Management Class Application Selection Panel for General Users.....	44
21. Management Class List.....	45
22. Delete Request Confirmation Panel.....	46
23. Management Class Sort Entry Panel.....	47

24. Management Class Display Panel.....	47
25. Source Configuration File Creation Request Confirmation Panel.....	48
26. Management Class Define Panels.....	49
27. Management Class Alter Panels.....	50
28. Configuration Application Selection Panel.....	54
29. Base Configuration Display Panel.....	55
30. Source Configuration File Creation Request Confirmation Panel.....	56
31. Base Configuration Define or Alter Panels.....	57
32. Source Configuration File Validation Panel.....	58
33. Browsing the Results of an ACS Routine Validation.....	59
34. When No Exits Are Defined.....	60
35. ACS Application Selection Panel.....	66
36. ACS Source Edit Panel.....	67
37. ACS Routine Translation Panel.....	68
38. Header for ACS Translator Listing.....	68
39. ACS Test Selection Panel.....	69
40. ACS Test Case Define/Alter Panel for DFSMS/VM.....	70
41. ACS Routines Test Panel.....	71
42. Header for ACS Test Listing.....	72
43. ACS Object Display Panel.....	72
44. ACS Object Delete Panel.....	73
45. Output Listing Disposition Panel.....	74
46. Equation to Determine the Migration Level 1 Space.....	76
47. Sample Selection Criteria Entries.....	88
48. Example of a Minidisk List Panel.....	90

49. Example of a Minidisk List Save Panel.....	91
50. Sample Entry to Save a List.....	92
51. Sample Entry to Replace a Saved List.....	92
52. Entering the MOVE Line Operator.....	99
53. Repeating the MOVE Line Operator.....	99
54. Using the MOVE Line Operator in Last-Use Mode.....	100
55. Move Entry Panel.....	101
56. Entering the MOVE List Command.....	103
57. Move Command Entry Panel.....	104
58. Move Confirmation Panel.....	105
59. Entering the CHECK Line Operator.....	107
60. Repeating the CHECK Line Operator.....	107
61. Using the CHECK Line Operator in Last-Use Mode.....	108
62. Check Entry Panel.....	109
63. Sample Check Request Results.....	109
64. Sample Check Output.....	110
65. The DFSMS Command Authorization Process.....	115
66. Example of the DFSMS MANAGE Command Output.....	119
67. Example of the DFSMS MIGRATE Command Output.....	121
68. Example of the DFSMS RECALL Command Output.....	122
69. Example of the DFSMS REPORT SPACEMANAGEMENT FILESPACE Command Output.....	124
70. Example of an Error Condition Displayed in the File Space Report File.....	125
71. Example of an Error Condition Displayed in the File Space Report File.....	125
72. Example of an Error Condition Displayed in the File Space Report File.....	125
73. Example of the DFSMS REPORT SPACEMANAGEMENT STORGROUP Command Output.....	126

74. Example of an Error Condition Displayed in the Storage Group Report File.....	127
75. Example DFSMS/VM HELP Panel.....	134
76. Example of the DFSMS ACTIVATE Command Output.....	135
77. Example of the DFSMS ALTER Command Output.....	138
78. Example of the DFSMS CHECK Command Output.....	140
79. Example of the DFSMS CONVERT Command Output.....	141
80. Example of the DFSMS COPY Command Output.....	144
81. Example of the DFSMS DELETE MIGRATION Command Output.....	146
82. Example of the DFSMS DELETE ML2BACKUP Command Output.....	146
83. Example of the DFSMS DISCARD Command Output.....	147
84. Example of the DFSMS MANAGE Command Output.....	149
85. Example of the DFSMS MIGRATE Command Output.....	151
86. Example of the DFSMS MOVE Command Output.....	156
87. Example of the DFSMS QUERY Command Output.....	158
88. Example of the DFSMS RECALL Command Output.....	159
89. Example of the DFSMS REPORT FILESPACE Command Output.....	161
90. Example of the DFSMS STOP Command Output.....	162
91. Example of Repeating a Line Operator.....	167
92. Using a Line Operator in Last-Use Mode.....	168
93. Using Last-Use Mode for Multiple Entries.....	168
94. Repeating a Line Operator in Last-Use Mode.....	169
95. Sample Entry to Initiate a User Extension as a Command.....	172
96. Sample Line Operator History for a User Extension.....	172
97. Sample Entry to Initiate a User Extension as a Line Operator.....	172
98. Repeating a User Extension Entered as a Line Operator.....	173

99. Saved ISMF Lists Panel After Completion of the Line Operator.....	173
100. Sample Code Segment.....	175
101. Using INCLUDE and EXCLUDE.....	186
102. Sample ACS REXX Routine for a Large System (Part 1 of 2).....	200
103. Sample ACS REXX Routine for a Large System (Part 2 of 2).....	201
104. Parameter Structure for the ACS Module Exit.....	205
105. Layout of ACS Module Exit Parameter List.....	208
106. Layout of IGDACERO Mapping Macro (Part 1 of 4).....	209
107. Layout of IGDACERO Mapping Macro (Part 2 of 4).....	210
108. Layout of IGDACERO Mapping Macro (Part 3 of 4).....	211
109. Layout of IGDACERO Mapping Macro (Part 4 of 4).....	212
110. Layout of IGDACERW Mapping Macro.....	213
111. Parameter Structure for the ACS Interface Routine.....	214
112. Sample ACS Routine for a Small System (Part 1 of 2).....	225
113. Sample ACS Routine for a Small System (Part 2 of 2).....	226
114. Sample ACS Routine for Large Systems (Part 1 of 2).....	227
115. Sample ACS Routine for Large Systems (Part 2 of 2).....	228
116. Sample ACS Routine with Migration to ML1 and ML2 (Part 1 of 2).....	229
117. Sample ACS Routine with Migration to ML1 and ML2 (Part 2 of 2).....	230
118. Sample EXEC to Automate Storage Group Management.....	231
119. Sample PRMTNINI EXEC Output.....	262
120. Sample DFSMS CHECK report.....	263
121. Sample DFSMS REPORT report.....	263

Tables

1. Examples of Syntax Diagram Conventions.....	xix
2. ISMF Scroll Commands and Parameters.....	23
3. ISMF Special Symbols.....	31
4. Management Class Attributes.....	41
5. MINIDISK_LINK_MODE_LIMIT Value Relationships.....	97
6. DFSMS Command Processor Return Codes.....	132
7. ACS Routine Return Codes.....	215
8. Values of ACS Read-Only Variables for Sample Files.....	219
9. Sample Files and Directories for the Large System.....	220
10. Sample Files for the Small Environment.....	221
11. Sample Dates for Files in the Large System.....	221
12. Sample Dates for Files in the Small System.....	223
13. Common Control Variables for the ISMF lists.....	233
14. Variables for a File List Entry.....	235
15. Column Variables for Minidisk List Panel.....	236
16. Variables for a Management Class List Entry.....	237
17. Column Variables for the Saved ISMF Lists Panel.....	238
18. Conversion Table for Column Names.....	239
19. Column Description of the USER Accounting Record.....	257
20. System Accounting Record Column Description.....	259

About This Document

This document describes the storage administration procedures for Data Facility Storage Management Subsystem/Virtual Machine (DFSMS/VM), an IBM® z/VM® feature that improves productivity by providing storage management for Shared File System (SFS) storage and by easing the task of moving minidisks from one physical device to another.

Part one of this document contains SMS conceptual information and DFSMS/VM introductory information. Part two contains information that guides you through DFSMS/VM storage management and minidisk management operations. Part three contains reference information, including DFSMS commands, ISMF, exit information, and a sample DFSMS/VM environment.

Intended Audience

This document is intended for system administrators responsible for managing system storage in a z/VM environment.





Syntax, Message, and Response Conventions

The following topics provide information on the conventions used in syntax diagrams and in examples of messages and responses.

How to Read Syntax Diagrams

Special diagrams (often called *railroad tracks*) are used to show the syntax of external interfaces.

To read a syntax diagram, follow the path of the line. Read from left to right and top to bottom.

- The  symbol indicates the beginning of the syntax diagram.
- The  symbol, at the end of a line, indicates that the syntax diagram is continued on the next line.
- The  symbol, at the beginning of a line, indicates that the syntax diagram is continued from the previous line.
- The  symbol indicates the end of the syntax diagram.

Within the syntax diagram, items on the line are required, items below the line are optional, and items above the line are defaults. See the examples in [Table 1 on page xix](#).



Table 1. Examples of Syntax Diagram Conventions	
Syntax Diagram Convention	Example
Keywords and Constants A keyword or constant appears in uppercase letters. In this example, you must specify the item KEYWORD as shown. In most cases, you can specify a keyword or constant in uppercase letters, lowercase letters, or any combination. However, some applications may have additional conventions for using all-uppercase or all-lowercase.	 KEYWORD 

Table 1. Examples of Syntax Diagram Conventions (continued)

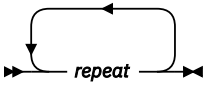
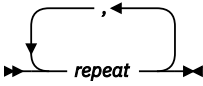
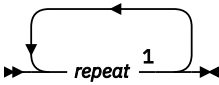
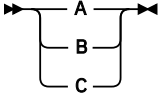
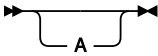
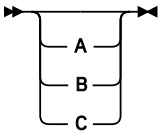
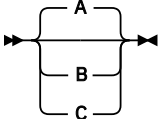
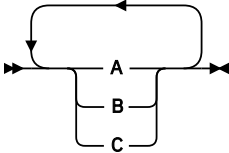

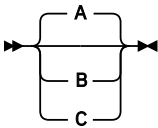
Syntax Diagram Convention	Example
Abbreviations <p>Uppercase letters denote the shortest acceptable abbreviation of an item, and lowercase letters denote the part that can be omitted. If an item appears entirely in uppercase letters, it cannot be abbreviated.</p> <p>In this example, you can specify KEYWO, KEYWOR, or KEYWORD.</p>	<p>►► KEYWOrd ◄◄</p>
Symbols <p>You must specify these symbols exactly as they appear in the syntax diagram.</p>	<p>* Asterisk</p> <p>: Colon</p> <p>, Comma</p> <p>= Equal Sign</p> <p>- Hyphen</p> <p>() Parentheses</p> <p>. Period</p>
Variables <p>A variable appears in highlighted lowercase, usually italics.</p> <p>In this example, <i>var_name</i> represents a variable that you must specify following KEYWORD.</p>	<p>►► KEYWOrd — <i>var_name</i> ◄◄</p>
Repetitions <p>An arrow returning to the left means that the item can be repeated.</p> <p>A character within the arrow means that you must separate each repetition of the item with that character.</p> <p>A number (1) by the arrow references a syntax note at the bottom of the diagram. The syntax note tells you how many times the item can be repeated.</p> <p>Syntax notes may also be used to explain other special aspects of the syntax.</p>	<p>  </p> <p>  </p> <p>  </p> <p>Notes: ¹ Specify <i>repeat</i> up to 5 times.</p>
Required Item or Choice <p>When an item is on the line, it is required. In this example, you must specify A.</p> <p>When two or more items are in a stack and one of them is on the line, you must specify one item. In this example, you must choose A, B, or C.</p>	<p>►► A ◄◄</p> <p>  </p>

Table 1. Examples of Syntax Diagram Conventions (continued)	
Syntax Diagram Convention	Example
<p>Optional Item or Choice</p> <p>When an item is below the line, it is optional. In this example, you can choose A or nothing at all.</p> <p>When two or more items are in a stack below the line, all of them are optional. In this example, you can choose A, B, C, or nothing at all.</p>	 
<p>Defaults</p> <p>When an item is above the line, it is the default. The system will use the default unless you override it. You can override the default by specifying an option from the stack below the line.</p> <p>In this example, A is the default. You can override A by choosing B or C.</p>	
<p>Repeatable Choice</p> <p>A stack of items followed by an arrow returning to the left means that you can select more than one item or, in some cases, repeat a single item.</p> <p>In this example, you can choose any combination of A, B, or C.</p>	
<p>Syntax Fragment</p> <p>Some diagrams, because of their length, must fragment the syntax. The fragment name appears between vertical bars in the diagram. The expanded fragment appears in the diagram after a heading with the same fragment name.</p> <p>In this example, the fragment is named "A Fragment."</p>	 <p>A Fragment</p> 

Examples of Messages and Responses

Although most examples of messages and responses are shown exactly as they would appear, some content might depend on the specific situation. The following notation is used to show variable, optional, or alternative content:

xxx

Highlighted text (usually italics) indicates a variable that represents the data that will be displayed.

[]

Brackets enclose optional text that might be displayed.

{ }

Braces enclose alternative versions of text, one of which will be displayed.

|

The vertical bar separates items within brackets or braces.

...

The ellipsis indicates that the preceding item might be repeated. A vertical ellipsis indicates that the preceding line, or a variation of that line, might be repeated.

Where to Find More Information

See [“Bibliography”](#) on page 269 at the back of this document.

Links to Other Documents and Websites

The PDF version of this document contains links to other documents and websites. A link from this document to another document works only when both documents are in the same directory or database, and a link to a website works only if you have access to the Internet. A document link is to a specific edition. If a new edition of a linked document has been published since the publication of this document, the linked document might not be the latest edition.

How to provide feedback to IBM

We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information. See [How to send feedback to IBM](#) for additional information.

Summary of Changes for z/VM: DFSMS/VM Storage Administration

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations for the current edition are indicated by a vertical line (|) to the left of the change.

SC24-6279-74, z/VM 7.4 (September 2024)

This edition supports the general availability of z/VM 7.4. Note that the publication number suffix (-74) indicates the z/VM release to which this edition applies.

SC24-6279-73, z/VM 7.3 (December 2023)

This edition includes terminology, maintenance, and editorial changes.

SC24-6279-73, z/VM 7.3 (September 2022)

This edition supports the general availability of z/VM 7.3. Note that the publication number suffix (-73) indicates the z/VM release to which this edition applies.

SC24-6279-01, for z/VM 7.2 (September 2020)

This edition supports the general availability of z/VM 7.2.

Part 1. System Managed Storage and DFSMS/VM

Chapter 1. System Managed Storage Concepts

System managed storage (SMS) is an approach to storage management that provides the ability to manage external storage for an enterprise. The tremendous growth in data and computer resources means that the task of managing data storage has become more complex, making the need for SMS evident. With system growth, installations have experienced some of the following problems:

- Shorter backup windows
- Rapid data growth without a corresponding increase in staff to help manage the growth
- Difficulty installing and using new storage technology efficiently
- Various amounts of inefficiently used or unused space
- No effective means to track data use (inactive data versus active data)

Although there are procedures for performing storage management, they are manual, labor-intensive operations. System managed storage software is designed to provide a consistent set of tools replacing the labor-intensive procedures of the past. The Data Facility Storage Management Subsystem (DFSMS) strategy directly responds to today's storage management requirements by providing:

- A high-speed data mover for facilitating installation of new DASD
- Storage management functions for efficiently handling inactive data by means of storage hierarchies
- Storage management reporting information
- Interactive Storage Management Facility (ISMF), a common interface for z/VM and MVS™ operating systems that performs storage management operations

The intent of this chapter is to give readers a conceptual understanding of why system managed storage is important and what DFSMS does for their installations.

Note: Some concepts covered in this chapter are currently only available in the MVS/ESA™ environment. The IBM Corporation has announced its intent to make available certain functions, when appropriate, for the z/VM environment.

What is System Managed Storage?

System managed storage is software that enables an installation to effectively manage external storage media with reduced staff and general user effort. DFSMS is IBM's strategic solution for managing external storage. *System managed* means the system manages active data, inactive data, performance, and security. Based on installation-defined requirements, the system matches the logical needs of the data to the physical characteristics of the storage devices and ensures device utilization. The storage administrator tailors system software to support the installation policies.

Goal of System Managed Storage

The goal of system managed storage is to reduce the amount of work involved with managing storage by:

- Reducing the labor involved in storage management by centralizing control, automating tasks, and providing interactive tools for storage administrators.
- Relieving general users of the details governing availability, space, and device management
- Providing efficient storage space utilization

Space Management Considerations for the Storage Administrator

When considering space management, it may be easier to understand problems and their solutions if you keep in mind that there are two different perspectives on this topic: the storage administrator's and the general user's.

The general user expects few or no processing failures, either in batch jobs or use of online systems. Failures consume time while general users investigate, understand, and then resolve problems. With this perspective, the questions a storage administrator should ask are:

- Is the space the users need available?
- What options should the users specify?
- What should the user do when an "out of space" condition occurs?

The answers to these questions minimizes failures and enhances performance. By ensuring that general users receive the space they need, the storage administrator reduces the possibility of failure. However, another objective is to make the best use of space. The storage administrator needs to ensure that a general user or a group does not abuse that resource (for example, creating "out of storage" conditions for other users), that the devices are being used efficiently, and that the overall amount of available space is adequate. Therefore, some other questions the storage administrator should ask are:

- How can I control allocation of space?
- How can I know when a file is no longer needed?

The use of space is enhanced by taking advantage of DFSMS. To maximize the benefits of centralized storage management, it is important to automate as many space management tasks as possible. DFSMS gives more control over storage by allowing assignment of management classes to files and directories. A management class contains a set of migration and expiration criteria that indicates how the storage administrator wants DFSMS to manage files. When a file meets the migration criteria of its management class, the file is migrated (compacted and moved) to secondary storage (secondary storage contains migrated data, see ["What is Secondary Storage?"](#) on page 6). When a file meets the expiration criteria of its management class, the file is erased.

Since migrated data is stored in a compressed format, DFSMS requires less space to store the data than it migrates from primary storage to secondary storage. When a general user references the migrated data, the data is returned to primary storage. Thus, DFSMS can be used to ensure that data is available to general users when they need it, without intervention, and at the same time use less storage for the data.

Availability Management

Managing the availability of data includes protecting the data against general user error by both providing the ability to recover the data to a previous state and by protecting the data against system and media failure. Categories of availability management are:

- Providing continuous availability for critical files
- Managing the backup process
- Managing the archive process
- Managing the vital records process
- Managing the disaster recovery process

Performance Management

Response time and job turnaround time are the standards by which general users judge performance. Managing performance, though, is a difficult task because of the many factors involved—processor time, the hardware configuration, where files are located in storage, and so forth. The placement of certain files can be controlled to provide a balance of performance across devices. These files include:

- Migrated files
- Certain application program libraries
- Database logs

SMS Constructs

SMS constructs consist of data class, storage class, management class and storage group items. By defining SMS classes, storage groups, and implementing automatic class selection (ACS), the system can be used to control an installation's storage resources. The data class, storage class, and storage group constructs are only available in DFSMS/MVS®. Both DFSMS/MVS and DFSMS/VM utilize the management class construct. In DFSMS/VM, the management class construct controls action for migration and expiration of files.

Storage Administration Groups

The storage administration group is responsible for:

- Providing storage management services that increase the productivity of general users
- Providing education and guidance to general users on storage-related issues
- Making efficient use of current storage resources
- Pursuing techniques that provide required levels of service in the most cost-effective manner
- Implementing plans that are compatible with the overall data processing center's direction
- Managing projects that relate to the storage subsystems

The Storage Administrator

The storage administrator is the coordinator of storage management for the enterprise. This person understands the enterprise needs, creates policies to meet those needs, and defines the policies to DFSMS. DFSMS in turn can manage the enterprise data according to the storage management policies. The storage administrator is also responsible for defining and managing service levels, educating general users in how to request what they need, and managing the physical storage subsystem.

Functions performed by the storage administrator include:

- Space management services
- Data availability services
- Storage subsystem performance tuning
- Reporting and accounting services
- Data security
- Migration to new or different devices

Storage Management in z/VM

In the z/VM minidisk environment, storage management is a laborious and inefficient task. The system programmer or system administrator allocates minidisks to a general user, and the general user manages that fixed amount of space. This approach has several limitations. First, minidisk space is unavailable for other applications, even though the space is unused. Next, when a general user runs out of space, the process of obtaining more space is either wasteful (for example, adding another minidisk), temporary (defining a temporary disk), or tedious (searching through files and deleting the old or obsolete). General users are forced to develop minidisk space management skills that usually have little to do with their jobs. Depending on installation policy about assigning new or increased minidisk space to general users, old and unwanted files may be retained for undesirable periods of time.

In today's z/VM environment, storage management is simplified, efficient, and centrally controlled. This advancement in storage management is due to IBM's Shared File System (SFS) and DFSMS/VM. SFS provides many functions to make storage management more efficient and addresses limitations of the minidisk file system. SFS eliminates wasted space within a minidisk and DASD fragmentation (the small number of cylinders left between minidisks). It allows an installation to assign a specified amount of storage space to a general user. This space is dynamically allocated (as needed) from a larger pool of space. Unlike the minidisk concept, this space is actually not allocated to a general user until the space

is required for storing data. DFSMS/VM helps minimize the amount of DASD containing inactive SFS files by keeping only active SFS files on the expensive, high-speed DASD, migrating inactive files to secondary storage, and erasing files that have met expiration criteria.

DFSMS/VM provides capabilities to help installations manage their data according to installation policies. These capabilities can be divided into three groups:

Management classes

Installation storage management policies are specified in management classes, which are applied to SFS files and directories. DFSMS/VM uses management classes when making storage management decisions. Two types of criteria are supported in management classes. An installation may choose to use either or both of the following:

- **File Expiration:** When does the file expire? Once DFSMS/VM determines that the data is no longer useful, the file can be erased by DFSMS/VM. This determination is based on the time elapsed since file creation, the time elapsed since the file was last referenced, or a combination of both.
- **File Migration:** When is the file eligible to be compressed and moved to secondary storage? Eligibility is established when the file has not been referenced for a certain period of time.

An installation can set up various management classes in order to meet the requirements of different data types.

ACS processing

ACS processing provides the mechanism for automatically assigning management classes to SFS files or directories. The installation tailors ACS processing such that the management class containing the appropriate criteria is assigned accordingly to files and directories. ACS processing is also used to convert files to a DFSMS/VM environment.

Configurations

A configuration is a collection of policies consisting of management classes and ACS processing that implements storage management for an installation.

What is Secondary Storage?

DFSMS/VM migrates data from primary storage, compresses it, and stores it into secondary storage. Secondary storage consists of a two level hierarchy: migration level 1 (ML1) and migration level 2 (ML2). Ideally, the ML1 SFS file space is located on DASD that is slower or costs less than DASD used for general user file space (primary storage). For example, primary storage may be located on IBM 3390 DASD and ML1 may be located on IBM 3380 DASD. The intent of ML2 is to provide tape support in the secondary storage hierarchy.

Chapter 2. DFSMS/VM Function Level 221

DFSMS/VM Function Level 221 is a tool for managing low-activity and inactive data on DASDs in z/VM environments. DFSMS/VM makes it easier to specify criteria to help manage data. DFSMS/VM relieves your professional personnel and general users from manual storage management tasks and improves DASD utilization by automatically managing space in SFS file pools. Thus, manual activities to maintain online storage space are reduced to a minimum. DFSMS/VM's primary functions are to help you with space utilization and management, and to migrate CMS minidisks to new DASDs.

DFSMS/VM includes:

- Minidisk management capabilities
- Space management capabilities
- An Interactive Storage Management Facility (ISMF), a tool for selecting these capabilities
- z/VM native support for the IBM 3495 tape library data server, refer to [*z/VM: DFSMS/VM Removable Media Services*](#)

The intent of the following sections are to give you an understanding of what DFSMS/VM is and what it can do for your installation.

Minidisk Management

Minidisk management with DFSMS/VM consists of moving and checking CMS minidisks. The primary function of minidisk management is to help you migrate CMS minidisks to new DASDs quickly, efficiently, and with minimal impact to your general users. You can also create and tailor lists of minidisks. After tailoring the list, you may use the list to submit requests for the following tasks:

- Moving minidisks from one location to another
- Checking the integrity of minidisks

Space Management

Space management with DFSMS/VM is the process of managing SFS files located in primary storage according to an installation's policies. DFSMS/VM provides file migration and file erasure for managing storage. File migration is the moving of inactive data from primary storage to secondary storage. File expiration is the erasure of data that is no longer required. DFSMS/VM automates these space management processes by creating an environment where you choose the criteria that controls the SFS files in your installation.

Management Class

You may have previously maintained your storage by running resource reports and manually copying, deleting, or moving files. DFSMS/VM automates the management of storage at the file level by using the management class. A management class indicates to DFSMS/VM how to manage files. The attributes of each management class can specify if, when, and how a file is migrated or erased. The criteria for these management class attributes depends on the logical characteristics of the data, not physically where the data resides. Only the storage administrator can define or alter a management class.

To list, display, define, or alter management classes, use the ISMF Management Class Application Selection panel.

Rules and Usage of Management Classes

A management class name is a name associated with a file or directory that is set by the installation's ACS processing. Only one management class is assigned to a file or directory.

The NULL management class is a special type of management class. If a file has a NULL management class, DFSMS/VM uses the files parent directories' (the parent directory is where the file resides) management class to make storage management decisions about the file. If the file's parent directory management class is also NULL, DFSMS/VM uses the system's default management class for making storage management decisions for this file.

The condition may also exist when there is no management class associated with a file or directory. DFSMS/VM cannot manage a file with no associated management class. When a file pool is first placed under DFSMS/VM direction, the files and directories in the file pool do not have management classes. ACS processing is used to assign management classes to them.

For information about assigning management classes see [“Assigning Management Classes” on page 42](#).

The following rules determine the management class that is used during migration or expiration processing for a file:

- If the file has a management class name, the attributes defined for the file's management class are used to manage the file.
- If the file has the NULL management class, the following rules apply.
 - If the file's parent directory has a management class, the attributes defined for the management class of the directory are used to manage the file.
 - If the file's parent directory has the NULL management class, the attributes of the system default management class are used to manage the file.
 - If the file's parent directory has no management class associated with it, the file is not managed by DFSMS/VM.
- If the file has no management class associated with it, the file is not managed by DFSMS/VM.

ACS Processing

ACS provides the mechanism for automatically assigning management classes to SFS files and directories. In DFSMS/VM there are three ways to implement ACS processing. You may choose to use any or all or none of them. The three types are:

ACS language

ACS may be implemented by writing an ACS routine in the ACS language (a high-level programming language). The ACS language source statements are translated by the ACS translator into object form that is used during ACS processing. The ISMF facility can be used to translate, validate, and test the ACS language routines. The ACS language is the same in DFSMS/VM as it is in MVS with new variables for z/VM. For additional information, see [Chapter 6, “Automatic Class Selection,” on page 63](#), and [Chapter 16, “ACS Language Reference,” on page 177](#).

ACS REXX exit

ACS processing can also be implemented by an installation-written ACS REXX exit. Using the ACS REXX exit allows you to use the set of operators, instructions, and built-in functions of REXX. For performance reasons, you will probably want to compile your REXX ACS exit with the IBM Compiler for REXX/370. For additional information see [Chapter 6, “Automatic Class Selection,” on page 63](#), and [Appendix A, “ACS REXX Exit,” on page 191](#).

ACS module exit

ACS processing can also be implemented with an installation-written ACS module exit. This ACS module exit is an assembled and GENMODed high level assembler or assembler H installation written program. Using the ACS module exit allows you flexibility that the ACS routine cannot provide; however, this flexibility can generally be achieved by the ACS REXX exit. Because of its complexity and the possibility of error, other methods should be considered before using the module exit. For additional information, see [Chapter 6, “Automatic Class Selection,” on page 63](#), and [Appendix B, “ACS Module Exit,” on page 203](#).

If you choose not to implement any of the above, ACS processing assigns the NULL management class to files or directories that do not have a management class.

There is a distinct difference that must be understood between a file or directory having no management class and having the NULL management class. DFSMS/VM cannot make a decision when it finds no management class assigned to a file or directory; however, the NULL management class is a valid management class that indicates to DFSMS/VM that it needs to look at the parent directory or system default for the management class to use for that file during storage management processing.

Configurations

A DFSMS/VM configuration is a collection of installation-defined management classes and optional installation-defined ACS information. This collection of information implements the storage management policies of the installation. You can use ISMF to create a configuration and save it in an SFS file. When discussing configurations, there are three terms to be familiar with: *base configuration information*, *source configuration*, and *active configuration*.

The base configuration information consists of:

- A description of the configuration
- The name of the default management class

The default management class that you specify in the base configuration information is used for a file that is assigned the NULL management class and is in a directory that is assigned the NULL management class. The characteristics of the default management class are then used to determine how to manage such a file during processing of a DFSMS MANAGE command.

- The specification of how ACS processing is handled when a file is created.

You can choose how to handle ACS processing for files through the base configuration information options for INVOKE ACS FOR FILE CREATION. The INHERIT option is used to determine the management classes at a directory level. However, if you need file level management classes, you may run ACS for each file and assign a distinct management class to each file.

There are two options for running ACS for each file; YES and DEFER. The YES option assigns a management class at file creation. The DEFER option assigns a management class the first time the file is processed by a MIGRATE, MANAGE, or CONVERT command. At this time, the file size information is available to ACS processing. Since running ACS takes processing time, performance is affected when running at the file level. With the YES option the performance effect occurs at file creation. Using the DEFER option allows you to postpone the ACS processing until you run a MANAGE or CONVERT, which could then be done during times of low activity. ACS processing does not allow the assignment of no management class to a file or a directory. The DEFER option can be used to specify that ACS processing is not invoked during file creation and that no management class is assigned when the file is created.

Files created while the INHERIT option is in effect are assigned the NULL management class and are managed by the directory or system management class. The INHERIT option only affects file creation.

The base configuration information is contained in both source configurations and active configurations.

A source configuration is saved in an SFS file and contains the base configuration information, an optional ACS routine, and the management class definitions. Any number of source configurations may be created and saved. The DFSMS or ISMF ACTIVATE command is used to make the contents of a source configuration the active configuration. The active configuration is used by DFSMS/VM to implement the installation's policies. During activation processing, a copy of the source configuration becomes the active configuration. An active configuration remains active until another source configuration is activated. Should either DFSMS/VM or the system stop, the active configuration is saved and used the next time DFSMS/VM is started.

Under ISMF, the storage administrator may display the contents of source configurations and the active configuration. This capability includes viewing the management class definitions and other information associated with the configuration. If the storage administrator determines that an active configuration is not meeting the needs of the enterprise, a source configuration can be created or modified and then activated. That source configuration then replaces the existing active configuration. DFSMS/VM does not have to be stopped to activate a new configuration.

Valid and Invalid Configurations

A source configuration can be either valid or invalid:

Valid

A source configuration is valid when:

- The base configuration information exists.
- The default management class definition exists.
- All management classes assigned by the ACS routine are defined in the configuration, if a management class ACS routine is specified in the source configuration file.

A source configuration is validated when it is saved. If it is valid, no further action is needed. Otherwise, the ISMF VALIDATE function may be run to obtain a report indicating errors in the configuration. Only a valid source configuration can be activated.

Invalid

A source configuration is invalid when any of the preceding conditions are not met. An invalid source configuration cannot be activated.

Initialization and Active Configurations

When you start DFSMS/VM, it initializes with the last successfully activated configuration. If such a configuration is not available, a warning is issued. Even without an active configuration, DFSMS/VM is still functional; however, commands that are dependent upon having an active configuration for information required during processing are not available. In this situation, for DFSMS/VM to be fully functional, you must activate a source configuration.

Note: If an active configuration is not available, files and directories may still be created in DFSMS/VM-managed file pools, but they will have no management class assigned. However, each creation attempts ACS processing which can affect performance.

Different Active Configurations on Systems that Share Data

Considerations: A problem may arise if two different active configurations are used on systems that share SFS data in global file pools. Consider the systems shown in [Figure 1 on page 10](#).

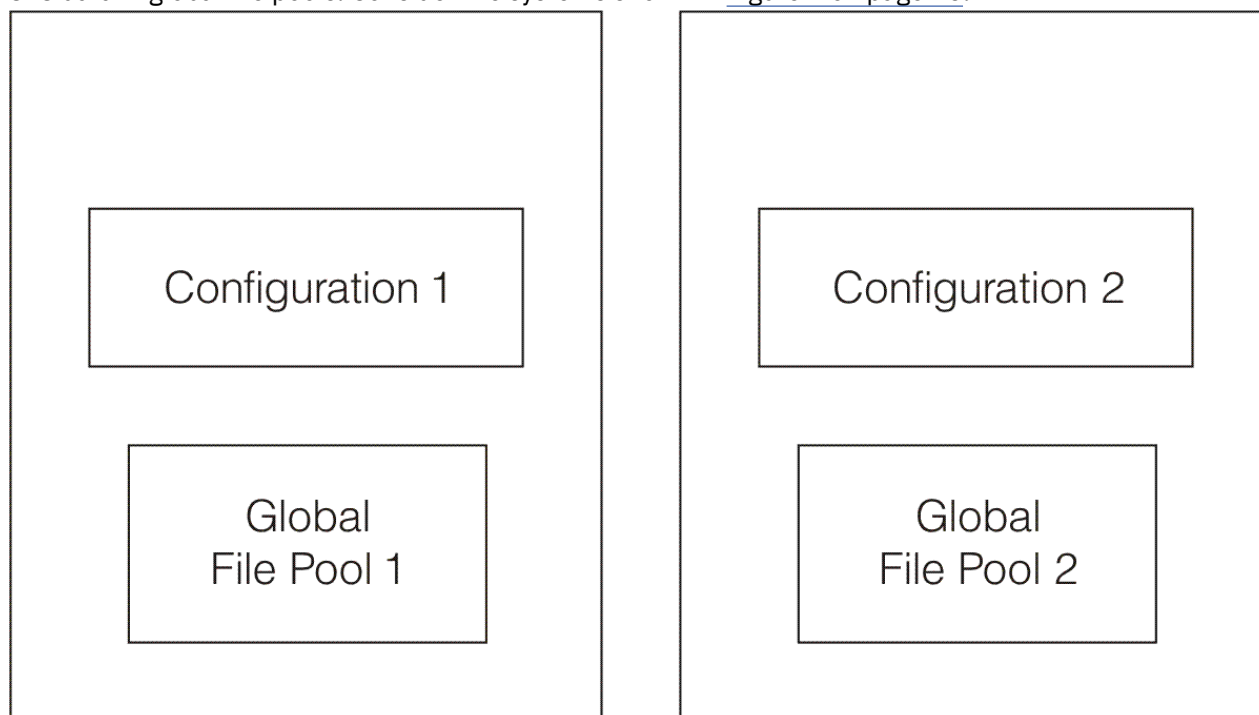


Figure 1. Example Environment with Configuration Use

In this example, the two systems each have a configuration and a file pool. The files created in *file pool 1* are assigned management classes from *configuration 1*, and files created in *file pool 2* are assigned management classes from *configuration 2*. However, general user dissatisfaction may occur if there are inconsistencies between the two systems. For example, files with the file type LISTING are erased one week after creation from z/VM System 1, but are never erased from z/VM System 2. For this reason, it is recommended that systems that share data with global file pools use the same active configuration. You can easily create the same active configuration by saving the desired source configuration in a global file pool and activating that source configuration on each system.

Another reason to use the same configuration in a multiple system environment is that you may only view the active configuration of the system you are currently logged on to. For example, suppose that you are working on z/VM system 2 and wish to look at the management class that is assigned to files stored in *file pool 2*. By using the management class application, you are able to review the characteristics of the management class contained in *configuration 2*. However, if you wish to look at the management class that is assigned to files stored in *file pool 1*, you must log on to z/VM system 1 and perform the same actions via the ISMF management class application as previously performed on z/VM system 2. If you attempt to review the management class on z/VM system 1 from z/VM system 2, the information provided comes from *configuration 2* not from *configuration 1*.

DFSMS/VM Control File

The DFSMS/VM control file (DGTVCNTL DATA) is in the VMSYS:DFSMS.CONTROL directory. This file contains the start-up parameters for DFSMS/VM. The control file is used to tailor DFSMS/VM. For additional information about the control file, see [z/VM: DFSMS/VM Customization](#).

Note: To make changes to the DFSMS/VM control file effective, you must stop and restart DFSMS/VM, see Chapter 11, “Starting and Stopping DFSMS/VM Processing,” on page 111.

The DFSMS/VM Interface

Two interfaces are available:

- The Interactive Storage Management Facility (ISMF)
- The DFSMS module from the CMS environment

ISMF provides a menu interface to the facilities of storage management for general users and storage administrators. See [“Understanding the Relationship of ISMF to DFSMS”](#) on page 20.

If you prefer a command-driven approach rather than a menu-driven approach for many of the DFSMS/VM operations, you can use the DFSMS command from the CMS environment. When using this interface, you have access to command-help and message-help facilities. See [Chapter 14, “DFSMS Commands,”](#) on page 131 for additional information about entering DFSMS commands and using the help facilities.

DFSMS/VM and the Shared File System

The Shared File System (SFS) is an extension of the CMS file system that allows CMS users to share data with greater flexibility and control than with minidisk sharing. SFS provides key functions needed by DFSMS/VM. First, SFS provides the *date of last reference (dolr)* on each file, which DFSMS/VM uses for tracking file usage. DFSMS/VM uses this information for determining when to migrate or expire files. Second, SFS does not physically use storage space until real data is stored (dynamic storage allocation). This differs from the minidisk structure, where a fixed amount of space is allocated and is not available to other applications.

The major benefit of using SFS is that it conserves system resources. Since space is only allocated when the general user needs it, the file pool only needs to be large enough to hold active user data. The efficient use of space by SFS also helps to conserve DASD space.

SFS uses the concept of a pool of storage instead of the traditional minidisk structure to manage the files of many general users. See [Figure 2](#) on page 12 for an illustration of how files are shared via a file pool in SFS. The pool of storage, referred to as a *file pool*, is composed of a number of minidisks managed

as one space. Within this space, the files of multiple general users are stored. There can be multiple file pools within an enterprise, each having the same structure, depending on the size and complexity of the enterprise.

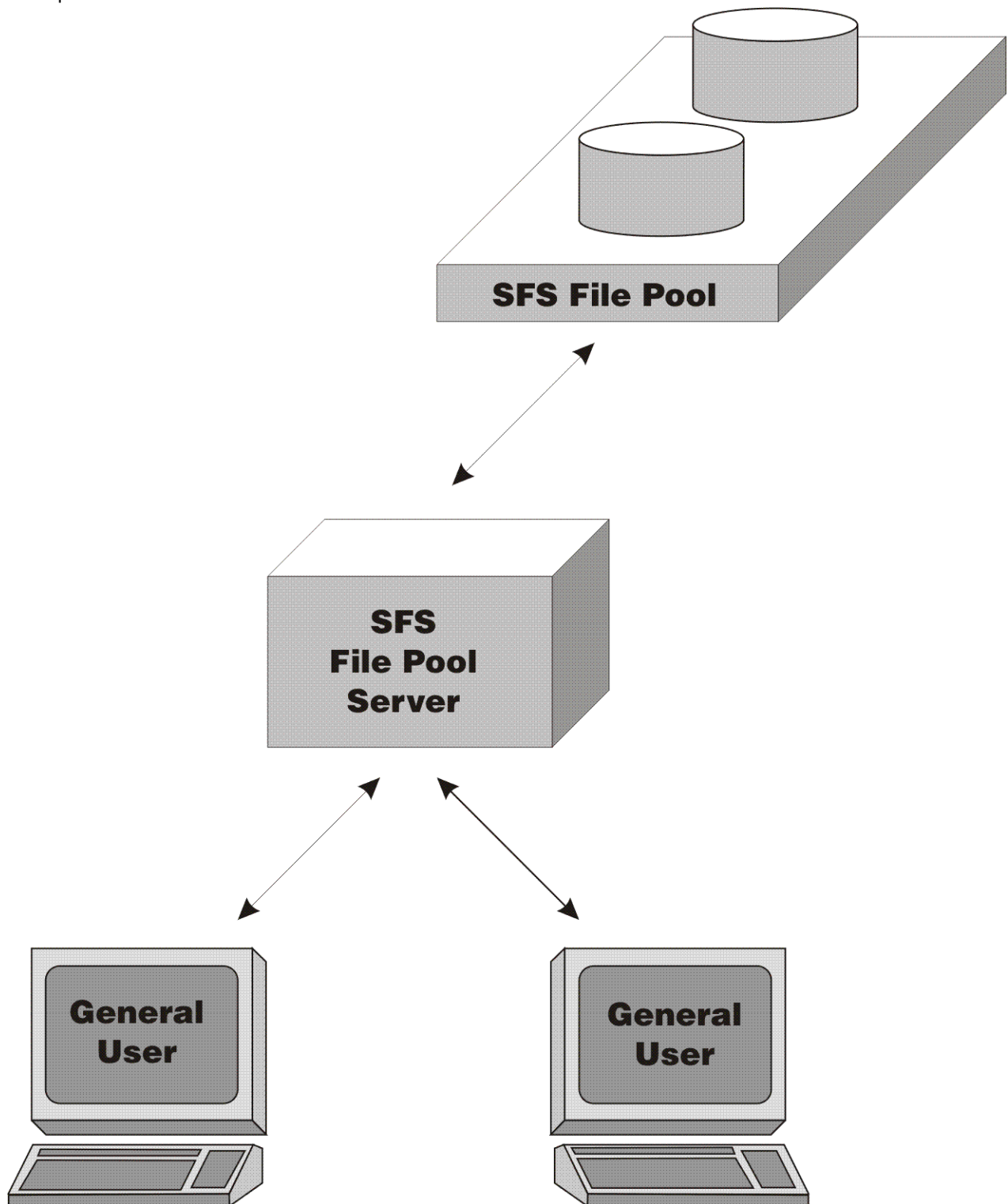


Figure 2. Shared File System

Within the file pool there are three basic data types: log data, control data, and user data. See [Figure 3 on page 13](#) for an illustration of the three areas in a file pool. The log data area contains two identical log minidisks that are used to provide data integrity in case of a failure. The control data contains information about the file pool. The user data area contains collections of minidisks defined as user *storage groups*. DFSMS/VM only manages the user data area. A general user would be enrolled in one of the storage

groups and would be allocated a specific amount of storage, defined in 4KB blocks. General users can only be enrolled in one storage group in a file pool but enrolled in many file pools.

File Pool Structure

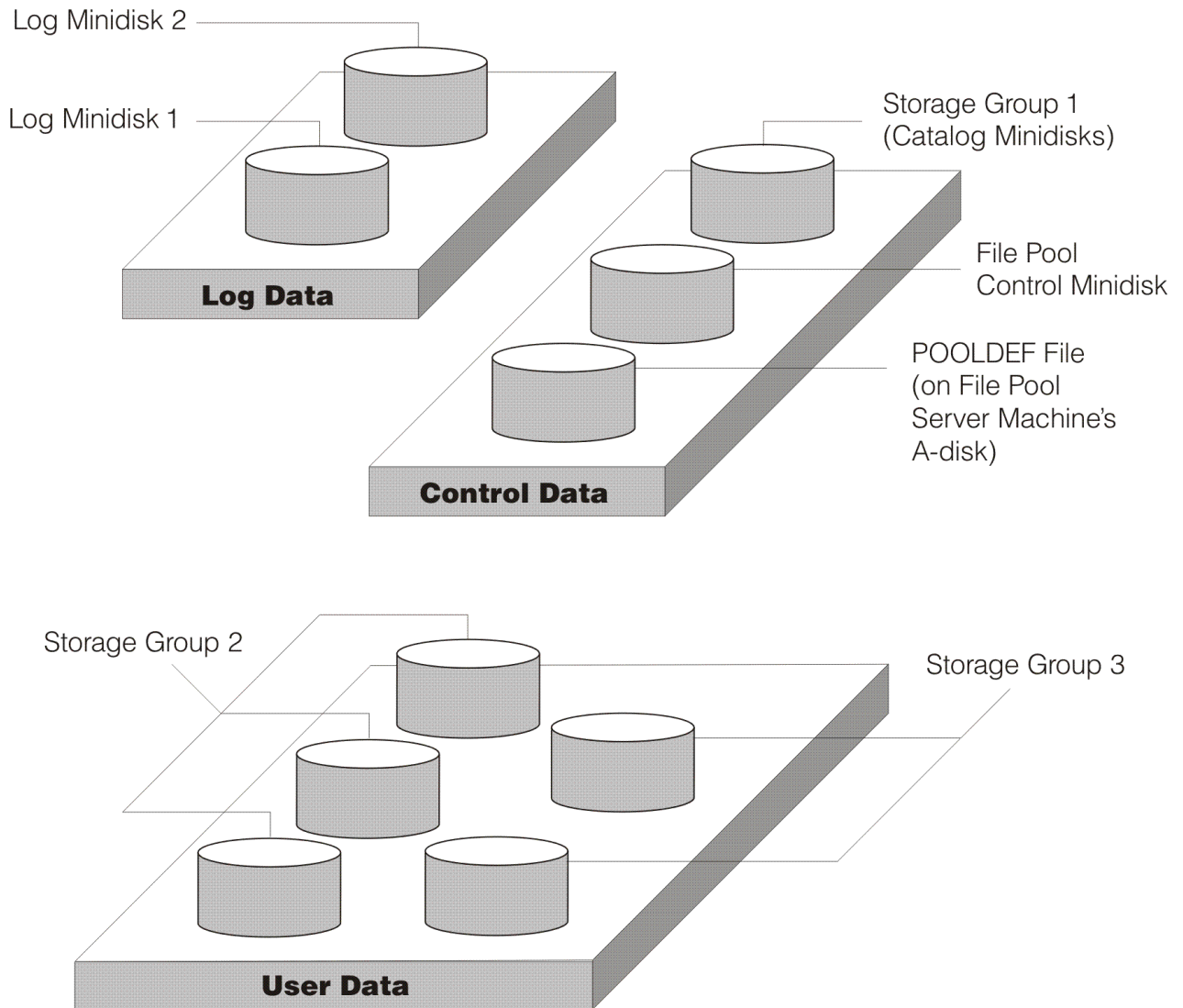


Figure 3. SFS File Pool Structure

SFS also provides general users with a new directory structure. See [Figure 4 on page 14](#) for an illustration of the SFS directory structure. Within an SFS directory, a general user can define subdirectories in a tree structure, which is more flexible than the minidisk structure for organizing files. Files can now be grouped together in subdirectories according to whatever criteria suits the general user.

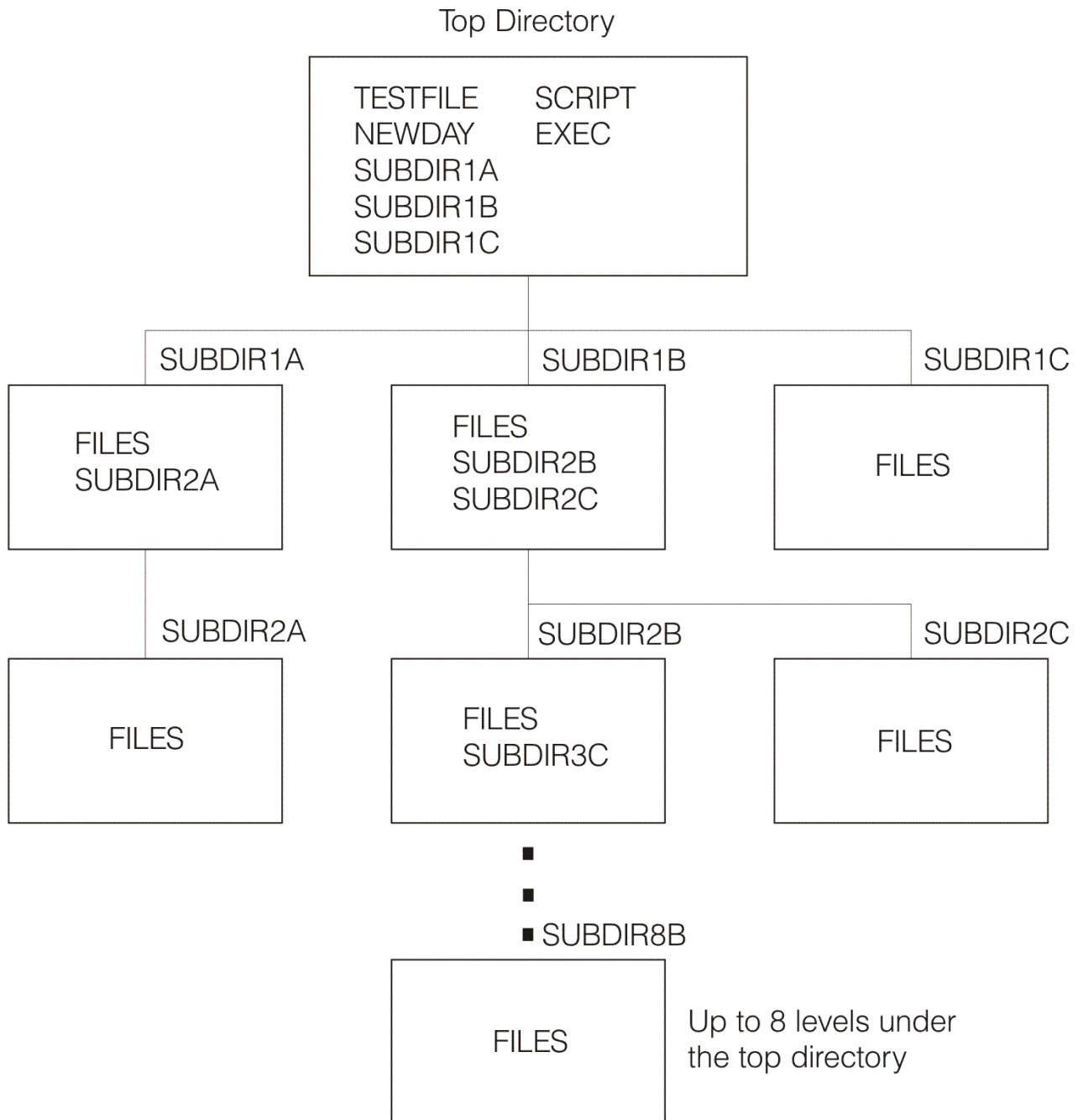


Figure 4. SFS Directory Structure

When general users are enrolled in the storage group, a *top directory* (user ID or file space) is defined for them. This directory can be used in the same way as a minidisk, but there is much more flexibility. The general user can define up to eight levels of subdirectories below the top directory. Because each directory level can contain files, general users can separate files into logical areas in their storage space. The name of each subdirectory can be up to 16 characters in length and the fully qualified directory name can be up to 153 characters in length. When working with files in directories, you can specify the fully qualified directory name, or you can use the CMS ACCESS command to access a directory as a file mode. When a directory is accessed as a file mode, it functions similarly to an accessed minidisk (that is, most actions that can be performed on an accessed minidisk can be performed on an accessed directory). The fully qualified name has the following format:

```
filepoolid:userid.dir1.dir2.dir3.dir4.dir5.dir6.dir7.dir8
```

where `filepoolid` is the name of the file pool (up to eight characters), `userid` (up to eight characters) is the name of the top directory (as well as the general user's user ID), and the *dirn* is the name of the directory at the *n*th level. For example:

```
VMSYSU:KGMITTON.STOCK.PRICES.SALES.PURCHASES.DIVIDENDS.GAINS.LOSSES.EXCHANGE
```

where the file pool ID is VMSYSU, and the top directory is KGMITTON followed by eight levels of directories.

You can also have *aliases* for files in SFS. The file that is the initial image is referred to as the *base file*. The alias has a complete file ID, but it does not take up any additional storage, so it can be thought of as a pointer to the base file. Also, the alias does not need to be in the same directory as the base file.

Note: When DFSMS/VM erases an expired file from SFS, it may erase either the entire file, or just the data of the file. This is determined by a parameter that you specify in the management class. By erasing just the data, DFSMS/VM leaves the alias and authorization structure intact.

The combination of DFSMS/VM and the Shared File System provides the tools for implementing system managed storage in your z/VM environment. DFSMS/VM provides further space usage efficiency with automatic class selection, data migration, recall, and expiration functions.

Shared File System References

For a list of publications that provide additional information about the Shared File System, see [“Bibliography” on page 269](#).

Part 2. Storage Administration Guide

Chapter 3. Using the Interactive Storage Management Facility

The Interactive Storage Management Facility (ISMF) provides an interface that allows you to use DFSMS/VM functions through a series of task-related menus, lists, and entry panels.

Help information is easily accessible online while you are working with the ISMF panels by typing **HELP** on the command line. For additional information about the ISMF help facility, see [“Getting Online Help While Using ISMF”](#) on page 32.

This chapter introduces features and characteristics unique to ISMF, provides details about moving around in the interface, and gives instructions for using the ISMF interface to start storage management tasks.

Entering and Exiting ISMF

The following sections discuss how to enter and exit ISMF within a CMS environment.

Entering ISMF

To invoke ISMF in a CMS environment, type **ISMF** or **DFSMS** at the command line and press ENTER. When you enter ISMF, the ISMF Primary Option menu (shown in [Figure 5 on page 19](#)) displays, including the window that shows IBM's copyright statement.

```
DGTSMD1          ISMF PRIMARY OPTION MENU
ENTER SELECTION OR COMMAND ==>

0  ISMF PROFILE    - Change ISMF user profile
                   - Control how ISMF logs and traces errors

1  FILE            - Access and view attributes of SFS files and
                   - directories
                   - Browse, edit, migrate, or recall SFS files
                   - Create and modify lists of SFS files and
                   - directories

3  MANAGEMENT CLASS - Display the attributes of a management class
                   - Create and modify lists of management classes

-----
| 5684-112 (C) Copyright IBM Corp. 1989, 1991. | rt, and erase
| All Rights Reserved.                         | log and list defaults
| US Government Users Restricted Rights -      |
| Use, duplication or disclosure restricted by |
| GSA ADP Schedule Contract with IBM Corp.   |
| Licensed Materials - Property of IBM.       |
|-----|
```

Figure 5. Primary Option Menu for General Users (with Copyright Window)

Once you have entered the ISMF interface, you can select the tasks you want to perform. For information on the tasks available within ISMF, see [“Understanding the Relationship of ISMF to DFSMS”](#) on page 20 and [“Selecting an ISMF Task”](#) on page 23.

Exiting ISMF

To end an ISMF session, return to the ISMF Primary Option menu by using the END or RETURN command. Then, exit ISMF in one of the following ways:

- Enter **X** on the command line and press ENTER.
- Press PF3 to display a Disposition of Console panel (ISPPFT01, ISPPFT02, ISPPFT03, or ISPPFT04). Fill in the **CONSOLE PROCESS OPTION** field and then press ENTER.

Understanding the Relationship of ISMF to DFSMS

ISMF is an interface to DFSMS. ISMF assists the storage administrator in utilizing storage management functions provided by DFSMS. You can use ISMF to accomplish some tasks that you cannot perform from the DFSMS command line. For instance, you can perform the following tasks by using ISMF to:

- Define, alter, copy, and display a management class
- Define and validate configurations
- Translate, validate, and test ACS routines
- Move a group of minidisks
- Check the integrity of minidisks owned by other users
- Create and save lists of files and directories, minidisks, and management classes
- Operate on groups of files in a list with commands such as MIGRATE and RECALL
- View previously saved lists

ISMF also provides many benefits:

- Panels of information that guide you through a task
- Online help providing descriptive information and an online index providing reference information
- Data entry fields primed with default values and the ability to set default values

However, some tasks cannot be accomplished by using ISMF. The following are tasks that can only be completed by issuing DFSMS commands on the CMS command line:

- Delete a file from ML1 or ML2 by issuing the DFSMS DELETE command
- Initiate DFSMS processing for files in a DFSMS managed filepool by issuing the DFSMS MANAGE command
- Obtain status reports on ML1 or ML2 by issuing the DFSMS REPORT command
- Assign management classes to files and directories by issuing the DFSMS CONVERT command
- Obtain the status of the DFSMS machines by using the DFSMS QUERY STATUS command

Working in the ISMF Environment

The ISMF environment is shaped by the Interactive Systems Productivity Facility (ISPF) presentation tool that was used to create ISMF. ISPF creates environments made up of panels that you move through to accomplish a given task.

Types of ISMF Panels

A panel is one screen of information and data entry fields displayed on a computer monitor. How the information and fields are organized indicates the type of panel. ISMF includes the following types of panels:

menu panel

where you select from a series of choices

selection panel

where you specify criteria in fields to select which data objects are displayed in a list panel

list panel

where you can view the characteristics of different types of data objects, such as an SFS file or a management class, in a tabular format

data entry panel

where you enter values in fields

display panel

where you only view displayed information

confirmation panel

where you answer a yes or no question

For specific information about panel formats, request help from any place in the ISMF interface.

All ISMF panels are formatted for use by the ISPF Dialog Manager. The standard screen for ISMF panels is 24 lines deep and 80 characters wide.

Identifying an ISMF Panel

Each ISMF panel has a unique panel ID. To display the panel ID, enter **PANELID ON** in the command line field of any ISMF panel. To stop the display of the panel ID, enter **PANELID OFF** in the command line field of any ISMF panel. Once you display the panel ID on a panel, it continues to display on all subsequent panels that you view, until you specify otherwise.

Initiating Action from an ISMF Panel

You can enter values in uppercase, lowercase, or mixed-case characters. However, ISMF processes all values as if they were upper case. For example, you cannot distinguish file names by case. If you name a file **STORAGE LIST** (upper case characters) and name a second file **Storage List** (upper and lower case characters), ISMF processes the second name in upper case and considers both formats as the same name for a single file.

ISMF saves some of the values you specify on a panel and displays these values the next time you enter that panel. However, changed values on ISMF selection and data entry panels are not preserved unless all values on the panel are valid and ENTER is pressed.

To initiate an action in ISMF, do one of the following:

- Type a value or a command at the command line and press ENTER
- Type a line operator name in the line operator data column of a list panel and press ENTER
- Fill the fields on the panel and press ENTER

Line operators can only be issued from a list panel and affect only one data object. Commands can be issued from any type of panel. Some commands when issued from the command line of a list panel affect the entire list of data objects.

For example, the MOVE command issued from the command line of the ISMF Minidisk List panel moves all the minidisks in the list. A MOVE line operator issued from the same panel against one minidisk, moves only that minidisk.

You can automate the issuing of commands by using an ISPF program function (PF) key. See [“Setting Program Function \(PF\) Keys”](#) on page 34 and [“Displaying Program Function Key Settings”](#) on page 33 for details.

Moving Through ISMF Panels

To move from panel to panel, use the following commands and keys.

Note: Check a panel’s return line to see if these commands are valid from that panel.

ENTER

Press ENTER to perform an action or make a selection

DOWN

Enter the DOWN command to view the next panel in a sequence of panels

END

Enter the END command to exit a panel or cancel a current task

Also, you can enter the RETURN command to go back to the ISMF Primary Option menu or, if RETURN is entered from within the Profile Application, to the ISMF Profile Option menu.

Moving Around ISMF List Panels

Most ISMF list panels contain more data columns and entries than can be displayed on one screen. You can view the additional data columns and entries by using scroll commands that are defined in the ISMF help facility. The ISMF scroll commands let you tailor what displays on a screen by enabling you to move vertically and horizontally through the list entries and data columns of an ISMF list panel.

To issue a scroll command, enter the command on the command line, then use the scroll parameter displayed in the SCROLL field or enter a new parameter. For example:

```
COMMAND ==> DOWN
```

```
SCROLL ==> HALF
```

When you issue a scroll command and ISMF performs the requested action, the value in the SCROLL field can change to indicate what portion of the list is currently displayed.

See [Table 2 on page 23](#) for the ISMF scroll commands and their parameters. The table also tells you whether the commands move through the data columns horizontally or through the list entries vertically. For more detailed information on the scroll commands and their parameters, see [“Using ISMF Commands and Line Operators” on page 162](#).

Table 2. ISMF Scroll Commands and Parameters		
Commands	Parameters	Movement
BOTTOM	None	Vertical
DOWN	1-9999, PAGE, HALF, MAX, CSR, DATA	Vertical
FIND	n= a data column number	Horizontal
LEFT	1-9999, PAGE, HALF, MAX, CSR, DATA	Horizontal
RIGHT	1-9999, PAGE, HALF, MAX, CSR, DATA	Horizontal
TOP	None	Vertical
UP	1-9999, PAGE, HALF, MAX, CSR, DATA	Vertical

Note: LEFT, RIGHT, and FIND are not valid from the Saved ISMF Lists panel. All other scroll commands are valid from any ISMF list panel.

Selecting an ISMF Task

When you enter ISMF, you will see the ISMF Primary Option menu, as shown in [Figure 5 on page 19](#). This menu lists the tasks that a general user can perform using ISMF. As a storage administrator, your first task is to change your profile so that you can view the list of tasks available to a storage administrator.

Changing the User Profile

To change your profile, follow these steps:

1. Select option 0 on the ISMF Primary Option menu and press ENTER. The menu entitled ISMF Profile Option menu (shown in [Figure 6 on page 23](#)) displays.

```

DGTSPPF1          ISMF PROFILE OPTION MENU
ENTER SELECTION OR COMMAND ==>

SELECT ONE OF THE FOLLOWING:

  0  USER MODE          - Change ISMF user mode
  1  LOGGING CONTROL     - Control how ISMF logs and traces errors
  X  EXIT                - Return to ISMF Primary Option Menu

USE ENTER TO SELECT OPTION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 6. ISMF Profile Option Menu

2. Select option 0 on the ISMF Profile Option menu and press ENTER.
3. Change the value in the USER MODE field to **2** and press ENTER.

4. Type END on the command line and press ENTER. Repeat this step four times, and you will arrive at the ISMF Primary Option menu for storage administrators as shown in [Figure 7 on page 24](#).

```
DGTSMMD2          ISMF PRIMARY OPTION MENU
ENTER SELECTION OR COMMAND ==>

SELECT ONE OF THE FOLLOWING:

 0 ISMF PROFILE      - Change ISMF user profile
                     - Control how ISMF logs and traces errors
 1 FILE              - Access and view attributes of SFS files and directories
                     - Create and modify lists of SFS files and directories
                     - Browse, edit, migrate, or recall SFS files
 2 MINIDISK           - Create and modify lists of minidisks
                     - Move and check minidisks
 3 MANAGEMENT CLASS - Define, alter, and display management class attributes
                     - Create and modify lists of management classes
 7 ACS                - Create, translate, validate, test, display, and
                     - delete Automatic Class Selection routines
 8 CONFIGURATION      - Display or alter configurations
                     - Define, validate and activate a base configuration
 L LIST               - Access, print, filter, sort, and erase previously
                     - saved lists
 X EXIT               - End ISMF using console, log and list defaults

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 7. Primary Option Menu for Storage Administrators

With the ISMF Primary Option menu for storage administrators displayed, you can view all the tasks available within ISMF. The tasks are categorized into the following ISMF-defined groups:

- Profile Application
- File Application
- Minidisk Application
- Management Class Application
- Automatic Class Selection Application
- Configuration Application
- List Application

Profile Application

The Profile Application allows you to control your ISMF environment. For example, when you change the user mode to storage administrator mode, you are performing a Profile Application task. You can also use this application to specify what type of error data you want ISMF to record during your session. By default, ISMF records the following information in the log:

- Information about the processing of line operators and commands that can help you keep track of ISMF tasks you perform during a session
- Error conditions detected during processing to provide you with information useful for problem determination
- Information about any CP or CMS commands you issue from ISMF panels

Note: ISMF stores the profile information in the first available R/W filemode in the search order. Releasing this filemode while running ISMF can cause unpredictable results.

To change the defaults for error recording, follow these steps:

1. Select option 0 on the ISMF Primary Option menu and press ENTER.
2. Select option 1 on the ISMF Profile Option menu and press ENTER.
3. Fill the fields on the Logging Control Entry panel to indicate what error information you want recorded during your ISMF session. For information about how to fill these fields, type HELP on the command line and press ENTER.

4. Type END on the command line and press ENTER. Repeat this step until you return to the ISMF Primary Option menu.

Note: ISMF records error data in the ISPF log. If the ISPF log is not enabled, ISMF does not record error data.

File Application

The File Application provides the ability to perform tasks against Shared File System (SFS) files. You can create and view a list of files, modify a list of files, or perform storage management tasks using the File Application.

Creating a List of Files

1. Select option 1 on the ISMF Primary Option menu and press ENTER. The File Application Selection Entry panel (page 1 of 3) displays (see [Figure 8 on page 25](#)).
2. Specify what you want to include in your list by entering criteria in the fields of the three File Application Selection Entry panels. For field descriptions and assistance in filling in the fields, request help from any of the File Application Selection Entry panels. To move between panels, follow the instructions in the return line at the bottom of each panel.

```
DGTDSJF1          FILE APPLICATION SELECTION ENTRY PANEL          Page  1 of 3

COMMAND ===>

SELECT SOURCE OF FILE LIST  ===> 2    (1 - saved list, 2 - new list)

1  REUSE A SAVED LIST:

    LIST NAME  ===>

2  GENERATE A NEW LIST FROM CRITERIA BELOW:

    FN FT ===> * *              (Fully or partially specified)

    FM/DIRID ===> *
        ===>
        ===>
                                (File mode letter or directory name)

USE ENTER TO PERFORM SELECTION; USE DOWN COMMAND TO VIEW NEXT SELECTION PANEL;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 8. File Application Selection Entry Panel (Part 1 of 3)

```
DGTDSJF2          FILE APPLICATION SELECTION ENTRY PANEL          Page  2 of 3

COMMAND ===>

TAILOR LIST GENERATION BY SPECIFYING A SINGLE VALUE OR
A RANGE OF VALUES:

                                REL OP  VALUE    AND/OR  REL OP  VALUE
                                -----  -
ALLOCATED SPACE                ===>
CREATION DATE                  ===>
LAST REFERENCE DATE            ===>
NUMBER OF FILE RECORDS        ===>
RECORD LENGTH                  ===>

USE ENTER TO PERFORM SELECTION; USE UP/DOWN COMMAND FOR OTHER SELECTION PANELS;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 9. File Application Selection Entry Panel (Part 2 of 3)

DGTDSJF3 FILE APPLICATION SELECTION ENTRY PANEL Page 3 of 3

COMMAND ==>

TAILOR LIST GENERATION BY SPECIFYING A SINGLE VALUE OR
A RANGE OF VALUES:

	REL OP	VALUE	VALUE	VALUE	VALUE
AUTHORITY	==>				
DIRECTORY OWNER	==>				
MANAGEMENT CLASS NAME	==> NE	DEFAULT			
MIGRATED FLAG	==> EQ	YES			
OWNER	==>				
RECORD FORMAT	==>				
TYPE	==>				

USE ENTER TO PERFORM SELECTION; USE UP COMMAND FOR PREVIOUS SELECTION PANEL;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 10. File Application Selection Entry Panel (Part 3 of 3)

The input fields on all pages of the File Application Selection Entry panel hold their values from one session to another. These fields are primed with the values that ISMF used the last time the File Application list was generated. If you do not wish to use the values from previous sessions, change or erase the input fields in the File Application Selection Entry panel. Use the CLEAR command to erase the values in the input fields. See the ISMF Help facility for details about the File Application Selection Entry panel and the corresponding input fields.

- After you have specified the selection criteria for the list you are creating, press ENTER, and the list displays. A sample list is shown in [Figure 11 on page 26](#).

DGTLGP81 FILE LIST PANEL

COMMAND ==>

SCROLL ==> HALF
Entries 1-4 of 4
Data Columns 8-10 of 17

DIRID: *

ENTER LINE OPERATORS BELOW:

LINE OPERATOR	FILE NAME	FILE TYPE	FM	REC FMT	RECORD LENGTH	CREATE DATE	LAST REF DATE
---(1)---	--(2)---	--(3)---	(4)	(5)	---(8)---	---(9)---	---(10)---
	DGTTABL	MACLIB	A1	F	80	2001/03/25	2001/03/25
	PROFILE	EXEC	A1	F	80	2001/03/25	2001/03/25
	TABLES	MACLIB	A1	F	80	2001/03/25	2001/03/25
	PAYROLL	DATA	A1	F	80	2001/04/01	2001/04/03
-----	-----	BOTTOM	OF	DATA	-----	-----	-----

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 11. File List Panel

Modifying a List in the File Application

A list can be modified to include only those files and directories that you specify. Once you have displayed a list, you can do any of the following tasks to modify the list:

- Filter the list to display only those entries that meet the criteria you specify. Type FILTER on the command line and press ENTER.
- Rearrange the entries in the list. Type SORT on the command line and press ENTER.
- Hide list entries. Type HIDE in the line operator column next to a file or directory and press ENTER.
- Redisplay hidden list entries. Type RESHOW on the command line and press ENTER.

- View the original list. Type REFRESH on the command line and press ENTER.

Performing Storage Management Tasks in the File Application

You can perform storage management tasks on the entire displayed list, or on individual entries within the list. For example, suppose you modified the list to include only those entries created before a certain date. Next you could migrate all the files in your modified list to a different storage medium. You can perform these storage management tasks on the displayed list:

- Save the list. Type SAVE *name* on the command line and press ENTER.
- Migrate a single file. Type MIGRATE in the line operator column next to the file you want to migrate. Press ENTER.
- Migrate a list of files. Type MIGRATE on the command line and press ENTER.
- Recall a file. Type RECALL in the line operator column next to the file you want to recall. Press ENTER.
- Recall a list of files. Type RECALL on the command line and press ENTER.
- Alter the management class of a file or directory. Type ALTER in the line operator column next to the file name and press ENTER.
- Alter the management class of the top-level directory. Type ALTER FILESPACE in the line operator column next to the directory name and press ENTER.

Minidisk Application

You can use the Minidisk Application to create, view, and save a list of minidisks. You can also modify a list of minidisks and perform storage management tasks for the minidisks.

Creating a List of Minidisks

Much like the File Application, you can specify criteria about the minidisks you want to include in your list. The Minidisk Selection panels (page 1 and 2) enable you to specify this criteria. To create a list of minidisks, follow these steps:

1. Select option 2 on the ISMF Primary Option menu and press ENTER.
2. The Minidisk Selection panel (page 1 of 2) will display and you can select option 2 to generate a new list.
3. Fill in the fields on page 1 and 2 of the Minidisk Selection panels to specify criteria for your list. Use the online help for descriptions of the fields you need to fill.
4. Press ENTER to display the list.

Chapter 9, “Working with a Minidisk List,” on page 87 contains more detailed information about creating, displaying, and saving minidisk lists.

Modifying a List of Minidisks

Once you have displayed a list, you can do any of the following tasks to modify the list:

- Filter the list to display only those entries that meet the criteria you specify. Type FILTER on the command line and press ENTER.
- Rearrange the entries in the list. Type SORT on the command line and press ENTER.
- Hide list entries. Type HIDE in the line operator column next to minidisk and press ENTER.
- Redisplay hidden list entries. Type RESHOW on the command line and press ENTER.
- View the original list. Type REFRESH on the command line and press ENTER.

Performing Storage Management Tasks in the Minidisk Application

As a storage administrator, you can move minidisks from one storage location to another. You can query the status of a minidisk move request. You can also issue a check request to access information about the

size, location, and structure of a minidisk. [Chapter 10, “Moving and Checking CMS Minidisks,” on page 93](#) provides detailed information to help you complete the tasks of moving minidisks, checking minidisks, and querying minidisk operations.

Management Class Application

The ISMF Management Class Application allows you to:

- Define and alter a management class
- Display management class attributes
- Create and save a list of management classes

Beginning a Management Class Application Task

To begin any Management Class Application task, type **3** on the command line of the ISMF Primary Option menu for storage administrators (shown in [Figure 7 on page 24](#)). Then press ENTER.

After selecting 3 from the ISMF Primary Option menu, the Management Class Application Selection panel displays (see [Figure 12 on page 28](#)).

```
DGTSCMC2          MANAGEMENT CLASS APPLICATION SELECTION PANEL
COMMAND ===>

TO PERFORM MANAGEMENT CLASS OPERATIONS, SPECIFY:

CONFIGURATION FN FT ===> CONFIG SAMPLE      (File name and type or 'ACTIVE')
FM/DIRID ===> A
===>
===>                                     (File mode letter or directory name)

MANAGEMENT CLASS NAME ===> *                (For Management Class List, fully or
                                           partially specified or * for all)

SELECT ONE OF THE FOLLOWING OPTIONS ===> 1
1 LIST      - Generate a list of Management Classes
2 DISPLAY   - Display a Management Class
3 DEFINE    - Define a Management Class
4 ALTER     - Alter a Management Class

IF OPTION 1 IS CHOSEN ABOVE,
RESPECIFY SORT CRITERIA      ===> N      (Y or N)

USE ENTER TO PERFORM SELECTION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 12. Management Class Application Selection Panel for Storage Administrators

Understanding the Management Class Application Selection Panel

You can choose one of the following options from the Management Class Application Selection panel:

1 LIST

Creates a list of management classes and their attributes

2 DISPLAY

Views the attributes of only one management class

3 DEFINE

Specifies a management class and its attributes

4 ALTER

Modifies a management class and its attributes

Use the online help to obtain detailed information on how to enter values in the fields of the Management Class Application Selection panel. Also, see [“Management Class Application Selection Panel” on page 43](#) for additional information on management classes. However, you can use the following general guidelines to enter values in the Management Class Application Selection panel fields:

CONFIGURATION FN FT

Specifies the file name and file type of the source configuration file associated with the management classes you want to process, or specify the default 'ACTIVE' to indicate that you want to process those management classes associated with the active configuration.

FM/DIRID

Specifies the file mode or SFS directory name of the source configuration file specified in the CONFIGURATION FN FT field.

MANAGEMENT CLASS NAME

Specifies the name of the management class you want to process. You can partially specify a management class name through LIST; option1.

SELECT ONE OF THE FOLLOWING OPTIONS

Enter a number from 1 to 4 to select an option.

IF OPTION 1 IS CHOSEN ABOVE, RESPECIFY SORT CRITERIA

Enters **Y** or **N** to indicate that you want to specify new selection criteria for your management class list.

The primary tasks a storage administrator would do with the ISMF Management Class Application are define, alter or display a management class. For detailed instructions on defining, altering or displaying a management class, see [“Management Class Define or Alter” on page 48](#)

Automatic Class Selection (ACS) Application

ACS processing assigns a management class to a file or directory. You can use this application to prepare your ACS routines for processing. To prepare an ACS routine to assign management classes, do the following:

1. Edit the ACS routine to include the management classes you have defined for your environment. Option 1 on the ACS Application Selection panel allows you to edit an ACS routine.
2. Translate the ACS routine into object form using option 2 on the ACS Application Selection panel.

In addition, the following steps are recommended to prepare an ACS routine to assign management classes:

- Validate the source configuration file associated with your ACS routine by selecting option 3 on the ACS Application Selection panel.
- Test the ACS routine by selecting option 4 on the ACS Application Selection panel.

When you have completed these tasks, the ACS routine is ready to be used in ACS processing. You can also display information about an ACS object by selecting option 5 and delete an ACS object by selecting option 6 on the ACS Application Selection panel. For detailed information you can use to complete the ACS Application tasks, see [“Automatic Class Selection Application” on page 66](#).

Configuration Application

The ISMF Configuration Application provides the ability to automate your policies for managing your installations DASD space. The Configuration Application allows you to define, display, alter, validate, and activate your source configurations. You can also display information about the active configuration.

Working with Configurations

To work with configurations, you need to use the ISMF Configuration Application by beginning with the following steps:

1. Select option 8 (Configuration) on the ISMF Primary Option menu for storage administrators and press ENTER. The Configuration Application Selection panel displays (see [Figure 13 on page 30](#)).

```

DGTSBSA1          CONFIGURATION APPLICATION SELECTION PANEL
COMMAND ===>

TO PERFORM CONFIGURATION OPERATIONS, SPECIFY:

CONFIGURATION FN FT ===> AACONFG1 SAMP          (File name and type or 'ACTIVE')
FM/DIRID ===> A
===>
===>          (File mode letter or directory name)

SELECT ONE OF THE FOLLOWING OPTIONS ===> 1

1 DISPLAY - Display base configuration information
2 DEFINE  - Define a base configuration
3 ALTER   - Alter a base configuration
4 VALIDATE - Validate a source configuration
5 ACTIVATE - Activate a configuration

USE ENTER TO PERFORM SELECTION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 13. Configuration Application Selection Panel

2. Select one of the following options to perform your task:

1 DISPLAY

Shows the base configuration information of a source configuration file.

2 DEFINE

Defines the base configuration for a source configuration file or, if the file does not exist, create it and then define its base configuration.

3 ALTER

Changes the base configuration information for a source configuration file.

4 VALIDATE

Validates an entire source configuration file.

5 ACTIVATE

Copies the specified source configuration file into the active configuration file.

These options provide the storage administrator with the ability to manipulate source configurations to suit your policies for managing storage space.

Any option that you choose displays helpful panels that prompt you to fill in various fields with necessary data to accomplish your task.

For more detailed information, see [Chapter 5, “Configurations,” on page 53](#).

List Application

The List Application enables you to view lists that you saved while using other ISMF applications. When you select option **L** from the ISMF Primary Option menu, the Saved ISMF List panel (see [Figure 14 on page 31](#)) displays a list of saved lists created from any of the following applications: the Management Class Application, Minidisk Application, or the File Application.

For example, if you saved a list of minidisks while using the Minidisk Application, you can view that saved list using the List Application. You can identify the originating application by the identifier in column 3, *List Type*, of the Saved ISMF Lists panel (for example, in entry 1 of [Figure 14 on page 31](#), the list type is MGMTCLAS signifying that list name MNGCLS is a saved list from the Management Class Application).

Note: After building a list, a saved list is created by issuing the ISMF SAVE command. For information about this command, see [“Using ISMF Commands and Line Operators” on page 162](#).

To view a saved list using the List Application, type LIST in the line operator data column next to the saved list you want to view.

```

DGTLLA1                      SAVED ISMF LISTS PANEL

COMMAND ===>                      SCROLL ===> HALF

                                   Entries 1-3 of 3
                                   Data Columns 3-8 of 8

ENTER LINE OPERATORS BELOW:

    LINE      LIST      LIST      LIST      LIST      LIST      LIST ROW      LIST
  OPERATOR   NAME      TYPE      DATE      TIME      USER      COUNT      UPDATE
  ---(1)---  --(2)---  --(3)---  ---(4)---  --(5)---  --(6)---  --(7)---  --(8)---
          MNGCLS  MGMTCLAS  1993/01/03  12.19.39  USER01          3          0
          MINOWN  MINIDISK  1993/01/21  09.14.23  USER01          9          0
          PAYROLL  FILEAPPL  1993/01/27  12.20.53  USER01         20          0
  -----  -----  -----  -----  -----  -----  -----  -----
                                   BOTTOM OF DATA

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 14. Sample Saved ISMF Lists Panel

The tasks you can perform on a list viewed within the List Application, are determined by the application in which you saved the list. For example, if you saved a minidisk list, you can use all the commands and line operators that are available in the Minidisk Application.

Information about the tasks you can perform appears in a pop-up window when the list initially displays. For further information about tasks that can be performed on the list, type HELP on the command line and press ENTER.

If during list building, ISMF cannot complete the requested list, ISMF provides a partial list and generates special symbols (see Table 3 on page 31) in column 1 or 11 next to the file name. You can then continue working with the partial list.

Table 3. ISMF Special Symbols	
SYMBOL	GENERATED IN COLUMN 1 or 11 WHEN
-----	A value is not available.
--NONE--	There is no management class that will be used to process this file, or this directory has no management class.
??????	That a minidisk is defined as a whole volume or that ISMF cannot derive or display the value.
<<<<<<	Numeric value is less than one, but greater than zero.
>>>>>>	Value is too large to display.

Sharing Saved Lists

Follow these guidelines when accessing saved lists:

- You can share saved lists among users in your installation. If your ISMF session uses a MACLIB that stores lists belonging to other users, you can access those lists automatically during your ISMF session. You can also have the lists sent to your user ID, and put them in any of the MACLIBs you use.
- If your ISMF session uses several MACLIBs, they may contain lists that have the same list name. For lists with duplicate names, only the first occurrence is shown in the list of saved ISMF lists.
- You can write EXECs to access and extract information from saved ISMF lists. For more information about accessing a saved list through an EXEC, see [“Accessing a Saved List through EXECs” on page 173](#).
- Saved lists can be implemented through the ISPF table services; refer to the *ISPF Dialog Management Guide*.

Printing a List

Use the ISPF commands PRINT or PRINT-HI to print a currently displayed list of saved lists. You can do this before or after you tailor the list.

The PRINT command prints your list one screen at a time and saves the output in your ISPF list file. If your list exceeds one page, you must scroll to the next screen of entries and issue the PRINT command for each displayed page of entries.

The PRINT-HI command prints your list in the same way as PRINT, and also prints any highlighted characters in bold.

Getting Online Help While Using ISMF

The ISMF interface provides help information in three areas:

- Messages appearing on an ISMF panel
- Help for tasks that can be performed using ISMF
- Index information on ISMF tasks and commands

Help Messages

While you are in ISMF performing tasks, the screen may display short and informational, warning, or error messages at the top right corner of ISMF panels. By entering the MESSAGE line operator in the line operator field next to that entry, you can also display a short message, as shown in [Figure 15 on page 32](#), related to the last request issued against a particular list entry.

```
MINIDISK SELECTION PANEL      LIST DOES NOT EXIST
COMMAND ==>
SELECT SOURCE OF MINIDISK LIST ==> 1      (1 - saved list, 2 - new list)
```

Figure 15. Sample Short Message

When a short message appears, you can type HELP and press ENTER to display a longer, more explanatory message. [Figure 16 on page 32](#) shows a sample long message.

```
MINIDISK SELECTION PANEL      LIST DOES NOT EXIST
COMMAND ==>
The specified list does not exist because it was not saved
SELECT SOURCE OF MINIDISK LIST ==> 1      (1 - saved list, 2 - new list)
```

Figure 16. Sample Long Message

Enter HELP again to display a message help panel that provides a detailed explanation of the error along with a suggested action to resolve the error.

Task-Related Help Panels

The ISMF help facility provides online reference information to help you use ISMF panels and perform ISMF tasks. You can display task-related help in one of the following ways:

- If no messages are currently displayed on the panel, type HELP on the command line to display the ISMF help panel.
- If a message is displayed on the panel, press ENTER to clear the message from the screen and then type HELP.

Task-related help includes:

- Overview panels that contain general information about a panel or task

- Selection panels that allow you to choose topics for more information
- Task panels that provide instructions to perform specific tasks such as hiding and reshowing list entries
- Field description panels that provide detailed information about fields on an ISMF panel
- Command and line operator panels that provide instructions to use specific commands and line operators

Help Index

The index can be accessed from anywhere within the ISMF help facility. The help index enables you to select other help topics contained in the ISMF help facility.

To display the help index, type INDEX on the command line of any help panel. To exit the help index, type END on the command line. When you exit the index, the ISMF panel from which you requested help is displayed again.

Moving Through Help Panels

To move from panel to panel within the help facility, use the following keys and commands:

ENTER

Page sequentially through the help panels associated with a specific functional panel.

END

Return to the ISMF functional panel where you originally asked for help. You can also use the RETURN command.

UP

Return to the help selection list that includes this topic as a choice.

BACK

Return to the previously viewed help panel.

SKIP

Skip to the help panel for the next related topic. (In some cases, using SKIP is the same as pressing ENTER.)

INDEX

Display the index for the ISMF help panels.

The PF key settings for the ISMF Primary Option menu or the ISMF application that you are using when you request help are also the same PF keys used within the help facility. When you install ISMF, the default PF key settings for END are PF3 and PF15, and the default key settings for UP are PF7 and PF19.

For more information about PF keys, see [“Setting Program Function \(PF\) Keys”](#) on page 34 below.

Displaying Program Function Key Settings

The ISPF **PFSHOW** command displays the current PF Key settings at the bottom of your screen; enter the command on any ISMF panel.

Because PF key settings can be different for the ISMF Primary Option menu and each application, the displayed PF key settings change as you move between applications, based on your key assignments for each one. For example, when you are using the Minidisk Application, the PF key settings you chose for that application are displayed. If you select the Profile Application, the PF key settings shown change to those selected for the Profile Application.

To remove your current PF key settings from the bottom of your ISMF panels during a session, enter **PFSHOW OFF** on the command line.

Setting Program Function (PF) Keys

ISMF supports one set of PF key assignments for the ISMF Primary Option menu, and one set for each ISMF application. When you install ISMF, all ISMF applications have the common PF key default assignments shown in [Figure 17 on page 35](#).

Note: The default for PF key 12 varies between cursor and retrieve depending on which ISMF application you are using. [Figure 17 on page 35](#) reflects the ISMF Primary Option menu's PF key defaults.

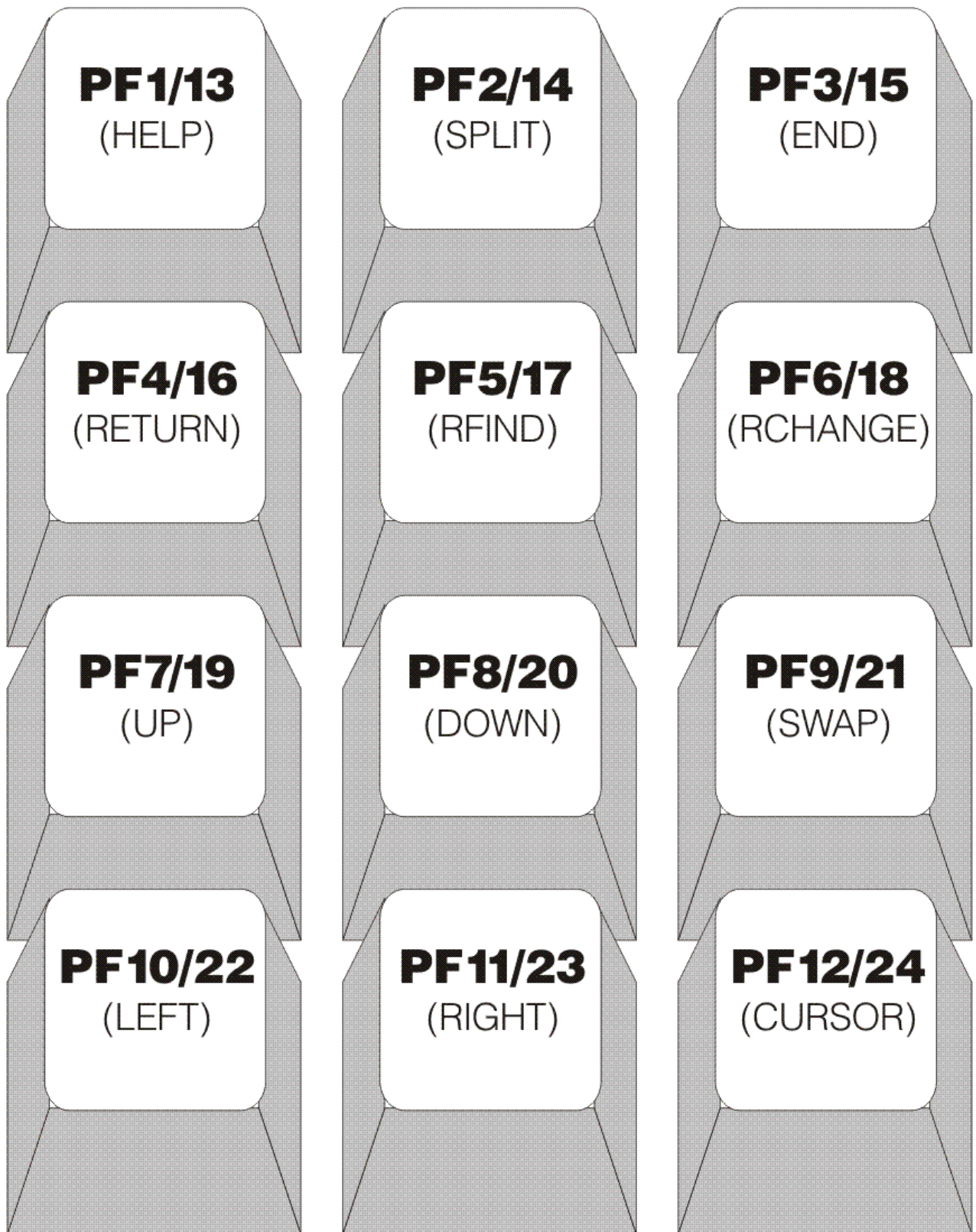


Figure 17. PF Key Defaults

Note: The indicated commands are ISPF system commands; for more information, see the *ISPF Dialog Management Guide*.

You can change the PF key settings at any time; for example, you may want to change some of the key assignments to other ISPF or ISMF functions or to ISMF commands or line operators that you frequently use.

You can use the KEYS command (supported by ISPF) to change your PF key assignments. Because key assignments are menu and application specific, your new PF key assignments are in effect only for the menu or for the application from which you issue KEYS.

The following examples provide instructions to change your PF key settings.

To change your PF key settings for the ISMF Primary Option menu:

- Display the ISMF Primary Option menu.
- Enter KEYS on the command line.

The PF Key Definitions and Labels—Primary Keys panel appears, (see [Figure 18 on page 36](#)). The panel shows the settings for PF keys 1 through 12.

```
----- PF KEY DEFINITIONS AND LABELS -----
COMMAND ==>

NUMBER OF PF KEYS ==> 12                TERMINAL TYPE ==> 3278

PF1 ==> HELP
PF2 ==> SPLIT
PF3 ==> END
PF4 ==> RETURN
PF5 ==> RFIND
PF6 ==> RCHANGE
PF7 ==> UP
PF8 ==> DOWN
PF9 ==> SWAP
PF10 ==> LEFT
PF11 ==> RIGHT
PF12 ==> CURSOR

PF1 LABEL ==>          PF2 LABEL ==>          PF3 LABEL ==>
PF4 LABEL ==>          PF5 LABEL ==>          PF6 LABEL ==>
PF7 LABEL ==>          PF8 LABEL ==>          PF9 LABEL ==>
PF10 LABEL ==>         PF11 LABEL ==>         PF12 LABEL ==>

Press ENTER key to save changes.  Enter END command to save changes and exit.
```

Figure 18. PF Key Definitions and Labels—Primary Keys Panel

- Enter your changes, and press ENTER.

For example, you might want to assign an ISMF command to one of the PF keys as shown below.

```
PF5 ==> ERTB   and   PF5 LABEL ==> ERR TABL
```

You can leave the label field blank if you want "ERTB" to display when you issue PFSHOW or you can enter a value in the label field to override the value in the PF key definition field. In this example, "ERR TABL" displays when PFSHOW is activated.

- When you have completed your changes, enter the END command or press PF3 to return to the ISMF Primary Option menu.

To change your PF key settings for an ISMF application:

- Select the application from the ISMF Primary Option menu. For example, you might select the Minidisk Application.
- Type KEYS on the command line of any panel within that application and press ENTER.

The PF Key Definitions and Labels panel appears; it is either the same or similar to the panel shown in [Figure 18 on page 36](#).

- Enter your changes, and press ENTER.

For example, you might want to assign an ISMF line operator to one of the PF keys. To do that, you must precede the name of the line operator with a colon (:). For example:

PF6 ===> :MOVE

- When you have completed your changes, enter the END command or press PF3 to return to the panel from which you issued the KEYS command.

To use a PF key that you have assigned to an ISMF line operator, position the cursor on the entry against which you want to perform the operation, then press the PF key. When you use a PF key assigned to an ISMF line operator, the line operator field must be blank before you press the PF key.

Chapter 4. Management Classes

The following sections describe management classes and explain a variety of tasks that can be accomplished when using the Management Class Application.

Contents of a Management Class

The management class is stored in configurations (for information about configurations see [Chapter 5, “Configurations,”](#) on page 53) and consists of the following:

- Management class name
- Description
- Expiration attributes
- Migration attributes

An installation defines as many management classes as needed to support the different types of data that an installation can have.

Management Class Name

A management class name consists of a 1–8 character name associated with a file or directory that is set by the installation's ACS processing. The characters supported in a management class name are any combination of upper-case alphanumeric characters and the national characters: "\$", "#", or "@". The first character of the management class name cannot be numeric.

Note: When specifying a management class name in ACS language or in the ISMF application you are not required to use uppercase characters. These two applications uppercase your input for you.

The management class name should describe the data rather than the management class criteria. For example, a management class name MIGAF9, meaning migrate files after nine days, is fine until migration needs to be changed to seven days. After the change to seven days the management class name of MIGAF9 is misleading causing you to convert all the files with that management class name to a new name.

Description

The 1–120 character description field is provided for you to describe your management class. Place any free-form information here that you or other storage administrators may need to know about the management class.

Expiration Attributes

One of the strengths of DFSMS is that it allows an installation to automatically erase files that are no longer needed. DFSMS provides three expiration attributes: EXPIRE AFTER NON-USAGE, EXPIRE SINCE CREATION, and EXPIRATION DISPOSITION. These attributes are used during DFSMS MANAGE command processing, and are ignored during DFSMS MIGRATE processing. The expiration attributes give you control of the amount of time a file associated with the management class is kept before it expires. When a file expires, it is eligible to be erased during MANAGE processing. When a file is erased, an expiration attribute also designates that either the entire file or just the data is erased. By erasing only the data, the alias and authorization structure is left intact. These attributes and values are specified by using the ISMF Management Class Application panels.

The EXPIRE AFTER NON-USAGE attribute specifies when a file can expire after it has not been read from or written to for some period of time. This is determined by the number of elapsed days (determined by the date-of-last-reference attribute) since the last file access. Valid values are:

nnnn DAYS

Specifies the number of elapsed days since the last file reference that must pass before the file expires. Valid values for **n** are 1–9999.

Note: The elapsed days are based on Greenwich Mean Time (GMT), not local time.

NOLIMIT

Specifies that a file is not eligible for expiration based on the date of last reference (when the file was last read or written to). If NOLIMIT is specified, the expire since creation attribute is used. This is the default.

The EXPIRE SINCE CREATION attribute specifies the number of days after file creation that the file expires. This attribute is based on the file creation date in contrast with the EXPIRE AFTER NON-USAGE attributes that is based on the file's last reference date. Valid values are:

nnnn DAYS

Specifies the number of elapsed days since file creation that must pass before the file expires. Valid values are 1–9999.

Note: The elapsed days are based on Greenwich Mean Time (GMT), not local time.

NOLIMIT

Specifies that a file never expires based on how many days have elapsed since the file was created. If NOLIMIT is specified, the expire since creation attribute is used. This is the default.

You can specify both EXPIRE AFTER NON-USAGE and EXPIRE SINCE CREATION. The following indicate the guidelines for using the attributes together:

- When both attributes are set to NOLIMIT, files never expire
- When one attribute is set to NOLIMIT, the other attribute indicates when files expire
- When both attributes have a value other than NOLIMIT, files expire only when both criteria are met

The EXPIRATION DISPOSITION attribute indicates that either the data is erased, retaining the file structure (DATAONLY) or the data and the file structure is erased (ALL). Valid values are:

DATAONLY

Specifies the base file data is erased, but existing file authorizations and aliases are retained. For example, suppose you are the owner of a file that a number of general users access to deposit data for a quarterly status report. The users access the file through aliases they created. After the quarterly status report is complete, the file is no longer needed. You could have DFSMS/VM erase the file data and file structure, but then a new report is needed for the next quarter and the contributing users, again, require file access. Also each user would have to create an alias to access that file. Instead, DFSMS/VM can erase only the data in the file (deallocating the space used by the data), leaving the file access structure intact.

ALL

Specifies the base file is erased, including information relating to the base file, authorizations, and aliases. This is the default.

Migration Attributes

There are three migration attributes: PRIMARY STORAGE NON-USAGE, SECONDARY STORAGE NON-USAGE, and SPACE MANAGEMENT TECHNIQUE.

These migration attributes allow you to specify:

- Whether migration is permitted for a file
- When a file is eligible to migrate from primary storage
- Whether a file is migrated to ML1 or directly to ML2
- When a file is eligible to move from ML1 to ML2

If general users have any infrequently used files but require quick access to these files when they do use them, you might consider excluding the files from migration to tape via the migration attributes. These attributes and values are specified by using the ISMF Management Class Application panels.

The SPACE MANAGEMENT TECHNIQUE attribute controls whether file migration occurs during DFSMS MANAGE or DFSMS MIGRATE command processing. The following lists the valid values:

MIGRATE

Specifies that either the DFSMS MIGRATE or DFSMS MANAGE commands can migrate files. When defining a new management class, the panel is primed with MIGRATE.

NONE

Specifies that file migration is not allowed by either the DFSMS MIGRATE or DFSMS MANAGE commands. When this value is used, the PRIMARY STORAGE NON-USAGE and SECONDARY STORAGE NON-USAGE fields need to be blank.

The PRIMARY STORAGE NON-USAGE attribute specifies when a file that resides on primary storage is eligible, when it has not been read from or written to for some period of time, to migrate during DFSMS MANAGE command processing. This is determined by the number of elapsed days (determined by the date-of-last-reference attribute) since the last file access. Valid values are:

nnnn DAYS

Specifies the minimum number of elapsed days (determined by the date-of-last-reference) since a file was last accessed in primary storage before it becomes eligible to migrate.

Note: The elapsed days are based on Greenwich Mean Time (GMT), not local time.

When defining a new management class, the panel is primed with 2. A blank must be specified when the SPACE MANAGEMENT TECHNIQUE field is set to NONE; otherwise valid values are 0–9999.

NOLIMIT

Specifies that a file is not eligible for migration during DFSMS MANAGE command processing; however, the file can still be migrated via the DFSMS MIGRATE command.

The SECONDARY STORAGE NON-USAGE attribute determines where in secondary storage the file resides after it is migrated. A blank must be specified when the SPACE MANAGEMENT TECHNIQUE field is set to NONE; otherwise valid values are:

nnnn DAYS

0 indicates that when the file migrates, it migrates directly to ML2. Values 1–9999 indicate that when the file migrates, it migrates to ML1 and remains in ML1 for the defined number of days before moving to ML2. When defining a new management class, the panel is primed with 5.

NOLIMIT

Specifies that a file is only migrated to ML1 and never moved to ML2.

Note: If ML2 is not defined in the control file, eligible files are migrated to ML1.

Summary of the Attributes

Table 4 on page 41 provides a summary of which management class attributes are examined during DFSMS MIGRATE and DFSMS MANAGE command processing.

<i>Table 4. Management Class Attributes</i>		
Attribute	MIGRATE command	MANAGE command
EXPIRE AFTER NON-USAGE		X
EXPIRE SINCE CREATION		X
EXPIRATION DISPOSITION		X
SPACE MANAGEMENT TECHNIQUE	X	X
PRIMARY STORAGE NON-USAGE		X

Table 4. Management Class Attributes (continued)		
Attribute	MIGRATE command	MANAGE command
SECONDARY STORAGE NON-USAGE	X	X

Assigning Management Classes

A management class name is assigned by:

- Automatic class selection (ACS) processing when:
 - An SFS directory is created
 - An SFS file is created, if ACS is invoked for file creation
 - An SFS file is recalled from ML1 or ML2
 - The DFSMS CONVERT command is processing
 - The DFSMS MANAGE command is processing a file or directory having no management class
 - The DFSMS MIGRATE command is processing a file or directory having no management class
- Using the DFSMS ALTER command or either the ISMF ALTER line operator or the ISMF ALTER FILESPACE command from the ISMF File Application for an existing SFS file or directory. ACS processing does not occur.

A NULL management class is assigned:

- When the base configuration data for INVOKE ACS FOR FILE CREATION is set to INHERIT
- If ACS processing assigns the NULL management class
- If ACS processing makes no assignment and the file did not already have a management class
- If an active configuration does not have an ACS routine, REXX exit, or module exit (this is true only if a management class has not previously been assigned)

DFSMS/VM is able to manage data when management classes are associated with each file. Since DFSMS/VM does not manage files with no management class, an installation needs to be aware that no management class is associated with a file under the following conditions:

- DFSMS/VM is installed but not active when a file or directory is created.
- DFSMS/VM is not installed when files or directories are created.
- An error in ACS processing occurs when recalling or creating a file.
- A file pool is not managed by DFSMS/VM.
- There is no active configuration.
- If the DEFER option is specified for INVOKE ACS FOR FILE CREATION in the base configuration information, ACS processing is not invoked at file creation time, and the file has no management class.

There are various ways in which ACS processing can be invoked to associate a management class with a file or directory. During DFSMS CONVERT and MANAGE processing, all files and directories with no management class assigned are changed through ACS processing to have an associated management classes. If the DFSMS MIGRATE command is invoked on a file that has no management class, ACS processing is invoked to associate a management class with that file before migrate processing begins. When a file is recalled, ACS processing is performed to determine whether the existing management class or a new management class should be assigned to the file. The existing management class name is retained if either an ACS routine is not provided in the active configuration and no ACS exits have been activated for that configuration, or ACS processing does not change the existing management class. If the resulting management class name is not valid or not defined in the active configuration (this can only occur with the ACS exits), the file is recalled, but does not have a management class.

Working with ISMF for Management Class Application Tasks

The following sections show some of the panels you can use to complete tasks under the Management Class Application. The panel fields are listed to give you an idea of the type of information required to complete a task. Further information is available through the online help facility in ISMF. Refer to [“Getting Online Help While Using ISMF”](#) on page 32 for information describing the help facility.

Management Class Application Selection Panel

The Management Class Application Selection panel, shown in Figure 19 on page 43, is used by the storage administrator to generate a list of management classes and to display, define, or alter a management class. The Management Class Application Selection panel, shown in Figure 20 on page 44, is used by general users to generate a list of management classes and to display a management class. To access the Management Class Application Selection panel, select option 3 on the ISMF Primary Option menu. The Management Class Application Selection panel is always primed with 1 to select the list option.

```
DGTSCMC2          MANAGEMENT CLASS APPLICATION SELECTION PANEL
COMMAND ==>

TO PERFORM MANAGEMENT CLASS OPERATIONS, SPECIFY:

CONFIGURATION FN FT ==> 'ACTIVE'          (File name and type or 'ACTIVE')
FM/DIRID ==>
      ==>                                (File mode letter or directory name)
      ==>

MANAGEMENT CLASS NAME ==> *              (For Management Class List, fully or
                                          partially specified or * for all)

SELECT ONE OF THE FOLLOWING OPTIONS ==> 1
1 LIST   - Generate a list of Management Classes
2 DISPLAY - Display a Management Class
3 DEFINE - Define a Management Class
4 ALTER  - Alter a Management Class

IF OPTION 1 IS CHOSEN ABOVE,
RESPECIFY SORT CRITERIA      ==> N      (Y or N)

USE ENTER TO PERFORM SELECTION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 19. Management Class Application Selection Panel for Storage Administrator

The possible options are:

- Option 1 (LIST): This option generates a management class list with information about the management classes in a configuration. Within the Management Class List application, various list tailoring capabilities are available.
- Option 2 (DISPLAY): This option generates a panel used to display the attributes of one management class in formatted, nontabular form.
- Option 3 (DEFINE): This option is used to define the attributes of a new management class.
- Option 4 (ALTER): This option is used to alter the attributes of an existing management class.

```

DGTSCMC1          MANAGEMENT CLASS APPLICATION SELECTION PANEL
COMMAND ===>

TO PERFORM MANAGEMENT CLASS OPERATIONS, SPECIFY:

CONFIGURATION FN FT ===> 'ACTIVE'          (File name and type or 'ACTIVE')
FM/DIRID ===>
      ===>
      ===>          (File mode letter or directory name)

MANAGEMENT CLASS NAME ===> *              (For Management Class List, fully or
                                          partially specified or * for all)

SELECT ONE OF THE FOLLOWING OPTIONS ===> 1

  1 LIST    - Generate a list of Management Classes
  2 DISPLAY - Display a Management Class

IF OPTION 1 IS CHOSEN ABOVE,
  RESPECIFY SORT CRITERIA      ===> N      (Y OR N)

USE ENTER TO PERFORM SELECTION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 20. Management Class Application Selection Panel for General Users

The possible options are:

- Option 1 (LIST): This option generates a management class list consisting of 10 columns of information about the management classes in a configuration file. Within the Management Class List application, various list tailoring capabilities are available.
- Option 2 (DISPLAY): This option generates a panel used to display the attributes of one management class in formatted, nontabular form.

Management Class List

Selecting option 1 (LIST) on the Management Class Application Selection panel provides you with a list of management classes, shown in [Figure 21 on page 45](#). If you saved a list of management classes, the List Application is used to access a panel containing a listing of your saved lists (see [“List Application” on page 30](#) for additional information).

The Management Class List is sorted by management class name in ascending order as the default, or in the order resulting from a specified sort sequence. An entry in the list consists of columns which can be viewed by scrolling right (by using the RIGHT command).

The Management Class List entries are displayed one page at time. To view different sections of the management class list, enter a scroll amount in the SCROLL field at the top of the panel. Enter PAGE, HALF, MAX, CURSOR or a specific number of entries to use with the UP, DOWN, LEFT, or RIGHT commands. There are a variety of optional line operators and commands for the Management Class List panel. For example, the ALTER, COPY, DELETE, and DISPLAY are samples of some of the line operators that may be used. (The general user is only authorized to issue the DISPLAY line operator.) For additional information about commands and line operators, see [“Using ISMF Commands and Line Operators” on page 162](#). All of the scrolling information and LINE OP information is available in detail in the online help. [Figure 21 on page 45](#) is an example of the Management Class List panel.

```

DGTLP21          MANAGEMENT CLASS LIST PANEL          SCROLL ==> HALF
COMMAND ==>          Entries 1-5 of 5
                      Data Columns 3-7 of 11

CONFIGURATION FN FT: SMPCNFGL CONFIG
FM: A DIRID: VMSYSU:ADMIN

```

ENTER LINE OPERATORS BELOW:

LINE OPERATOR ---(1)---	MGMTCLAS NAME --(2)---	EXPIRE NON-USAGE ---(3)---	EXPIRE SINCE CREATION ----(4)----	EXPIRATION DISPOSITION ----(5)----	PRIMARY STORAGE ---(6)---	SECONDARY STORAGE ---(7)---
	DEFAULT	NOLIMIT	NOLIMIT	ALL	2	5
	DEFAULTN	NOLIMIT	NOLIMIT	DATAONLY	2	100
	LDEFAULT	300	1460	DATAONLY	100	100
	NOMIGRAT	32	32	DATAONLY		
	DIRML2	NOLIMIT	NOLIMIT	ALL	0	0

```

DGTLP21          MANAGEMENT CLASS LIST PANEL          SCROLL ==> HALF
COMMAND ==>          Entries 1-5 of 5
                      Data Columns 7-11 of 11

CONFIGURATION FN FT: SMPCNFGL CONFIG
FM: A DIRID: VMSYSU:ADMIN

```

ENTER LINE OPERATORS BELOW:

LINE OPERATOR ---(1)---	MGMTCLAS NAME --(2)---	SECONDARY STORAGE ---(7)---	SPC MGMT TECHNIQUE ---(8)---	LAST MOD USERID ---(9)---	LAST DATE MODIFIED ---(10)---	LAST TIME MODIFIED ---(11)---
	DEFAULT	5	MIGRATE	CAROL	2001/09/23	15:26
	DEFAULTN	100	MIGRATE	GREG	2001/09/29	06:23
	LDEFAULT	100	MIGRATE	DAVE	2001/10/01	08:12
	NOMIGRAT		NONE	KATHY	2001/11/26	17:30
	DIRML2	0	MIGRATE	DICK	2001/12/16	10:53

Figure 21. Management Class List

The following describes the information displayed in the data columns of the Management Class List panel:

- **LINE OPERATOR (1):** This field is used to specify a line operator for use with the management class name displayed in the list and also serves as the feedback area for the line operator. ALTER, COPY, and DELETE are the primary line operators that you specify on this panel. Some of the common line operators that you may also use are DISPLAY, HIDE, MESSAGE, and others listed in [“Using ISMF Commands and Line Operators”](#) on page 162.
- **MGMTCLAS NAME (2):** One-to-eight character management class name.
- **EXPIRE NON-USAGE (3):** This field indicates how many days must elapse since the last file access until the file expires. A NOLIMIT value in this field indicates that the amount of time elapsed since the file was accessed is not used in determining whether the file expires.
- **EXPIRE SINCE CREATION (4):** This field indicates the minimum number of days after file creation until the file expires. A NOLIMIT value in this field indicates the amount of time elapsed since the file was created is not used in determining whether the file expires.
- **EXPIRATION DISPOSITION (5):** This field indicates that either the file data is erased leaving the file structure intact or the file data and file structure are erased. File structure includes existing file authorizations and aliases.
- **PRIMARY STORAGE (6):** This field indicates the minimum number of days files associated with the displayed management class must remain unaccessed in primary storage before they are eligible to migrate from primary storage to secondary storage. This eligibility is only checked during DFSMS MANAGE command processing.
- **SECONDARY STORAGE (7):** This field indicates three items about files associated with the displayed management class: when the value is 0, the file is migrated directly to ML2; if the value is 1–9999, the

file is moved to ML2 after spending the specified number of days in ML1; and if the value is NOLIMIT, the file does not migrate or move to ML2.

- SPC MGT TECHNIQUE (8): This field indicates whether files associated with the displayed management class are eligible for migration. The value of NONE indicates that migration is not allowed and MIGRATE indicates that migration is allowed.
- LAST MOD USERID (9): This field is the one-to-eight character user ID of the storage administrator who last modified this management class.
- LAST DATE MODIFIED (10): This field is the date the management class entry was last modified.
- LAST TIME MODIFIED (11): This field identifies the time the management class entry was last modified.

By entering the DELETE line operator against a management class name in the line operator column of the Management Class List panel, the management class is deleted from the source configuration file. However, before this occurs, the Delete Request Confirmation panel is displayed, as shown in [Figure 22 on page 46](#). This panel allows you to confirm whether you want to delete the management class. You cannot delete a management class from the active configuration.

You need to be cautious when deleting management classes. A management class currently used by files or directories should not be deleted from a source configuration if you, again, plan to activate this source configuration. If this deletion and activation occurs, the files and directories assigned the deleted management class cannot be DFSMS/VM managed until their management class is changed using CONVERT or ALTER.

```
DGTDMDL1          DELETE REQUEST CONFIRMATION PANEL
COMMAND ==>>

TO CONFIRM DELETION OF THE FOLLOWING ELEMENT:

ELEMENT NAME: SOCIAL
ELEMENT TYPE: MANAGEMENT CLASS
ELEMENT FN FT : EMPLOYE PAYROLL
FM: A DIRID: VMSYSU:KEN

SPECIFY THE FOLLOWING:
  PERFORM DELETION ==>> N          (Y or N)

USE ENTER TO PERFORM OPERATION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 22. Delete Request Confirmation Panel

Another action that you may perform is sorting the management classes. The Management Class Sort Entry panel (see [Figure 23 on page 47](#)) is used by the storage administrator and the general user to sort the Management Class List by specified fields. The sort order defaults to the MANAGEMENT CLASS NAME (2) in ascending order (A). You may access this panel by:

- Entering option 1 (list) and Y to respecify sort criteria on the Management Class Application Selection panel, as shown in [Figure 19 on page 43](#).
- Entering SORT on the command line of the Management Class List panel. A sample list panel is shown in [Figure 21 on page 45](#).

For information about ISMF commands and line operators that are commonly used on this panel, see [“Using ISMF Commands and Line Operators” on page 162](#).

```

DGTDCMC4          MANAGEMENT CLASS SORT ENTRY PANEL
COMMAND ===>

SPECIFY SORT CRITERIA FROM THE ATTRIBUTE NUMBERS LISTED BELOW:
  MAJOR FIELD ===> 2      MINOR FIELD 1 ===>      MINOR FIELD 2 ===>

SPECIFY A FOR ASCENDING OR D FOR DESCENDING SORT ORDER:
  MAJOR FIELD ===> A      MINOR FIELD 1 ===>      MINOR FIELD 2 ===>

(1) LINE OPERATOR
(2) MANAGEMENT CLASS NAME
(3) EXPIRE AFTER NON-USAGE
(4) EXPIRE SINCE CREATION
(5) EXPIRATION DISPOSITION
(6) PRIMARY STORAGE NON-USAGE
(7) SECONDARY STORAGE NON-USAGE
(8) SPACE MANAGEMENT TECHNIQUE
(9) LAST MODIFIED USERID
(10) LAST DATE MODIFIED
(11) LAST TIME MODIFIED

USE ENTER TO PERFORM SORT;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 23. Management Class Sort Entry Panel

Management Class Display

Either selecting option 2 on the Management Class Application Selection panel or entering the DISPLAY line operator on a Management Class List panel will display the attributes of one management class in formatted, nontabular form. This information is displayed on the Management Class Display panels (see [Figure 24 on page 47](#)).

```

DGTICMC1          MANAGEMENT CLASS DISPLAY PANEL          Page 1 of 2
COMMAND ===>

CONFIGURATION FN FT: KEEP FILE
FM: A DIRID: VMSYSU:VMTEST0.

MANAGEMENT CLASS NAME: ARCHIVE

DESCRIPTION: FILES REQUIRING RETENTION.

EXPIRATION ATTRIBUTES
EXPIRE AFTER NON-USAGE:      NOLIMIT    (DAYS)
EXPIRE SINCE CREATION:      NOLIMIT    (DAYS)
EXPIRATION DISPOSITION:      ALL

```

```

DGTICMC2          MANAGEMENT CLASS DISPLAY PANEL          Page 2 of 2
COMMAND ===>

CONFIGURATION FN FT: KEEP FILE
FM: A DIRID: VMSYSU:VMTEST0.

MANAGEMENT CLASS NAME: ARCHIVE

MIGRATION ATTRIBUTES
PRIMARY STORAGE NON-USAGE:      3    (DAYS)
SECONDARY STORAGE NON-USAGE:    5    (DAYS)
SPACE MANAGEMENT TECHNIQUE:      MIGRATE

```

Figure 24. Management Class Display Panel

The following are descriptions of the panel fields displayed in the Management Class Display panel:

- **MANAGEMENT CLASS NAME:** This field contains the name of the management class currently displayed.
- **DESCRIPTION:** This field contains the description (if any exists) of the displayed management class.

- **EXPIRE AFTER NON-USAGE (3):** This field indicates how many days must elapse since the last file access until the file expires. A NOLIMIT value in this field indicates that the amount of time elapsed since the file was accessed is not used in determining whether the file expires.
- **EXPIRE SINCE CREATION (4):** This field indicates the minimum number of days after file creation until the file expires. A NOLIMIT value in this field indicates the amount of time elapsed since the file was created is not used in determining whether the file expires.
- **EXPIRATION DISPOSITION:** This field indicates that either the file data is erased leaving the file structure intact or the file data and file structure are erased. File structure includes existing file authorizations and aliases.
- **PRIMARY STORAGE NON-USAGE:** This field indicates the minimum number of days files associated with the displayed management class must remain unaccessed in primary storage before they are eligible to migrate from primary storage to secondary storage. This eligibility is only checked during DFSMS MANAGE command processing. Specifying NOLIMIT restricts the file to migration by command.
- **SECONDARY STORAGE NON-USAGE:** This field indicates three items about files associated with the displayed management class: when the value is 0, the file is migrated directly to ML2; if the value is 1–9999 the file is moved to ML2 after spending the specified number of days in ML1; and if the value is NOLIMIT, the file does not migrate or move to ML2.
- **SPACE MANAGEMENT TECHNIQUE:** This field indicates whether files associated with the displayed management class are eligible for migration. The value of NONE indicates that migration is not allowed and MIGRATE indicates that migration is allowed.

Management Class Define or Alter

The Management Class Define and Alter panels are used to define attributes of a new management class or to change the attributes of an existing management class.

The following are various ways you may invoke the Management Class Define or Alter panels:

- Select Option 3 (Define a management class) on the Management Class Application Selection panel for an existing source configuration. If option 3 is specified for a source configuration file that does not exist, the Source Configuration File Creation Request Confirmation panel is displayed (see [Figure 25 on page 48](#)). If the storage administrator replies, on the Source Configuration File Creation Request Confirmation panel, that the file is to be created, the Management Class Define panel is displayed.

```

SOURCE CONFIGURATION FILE CREATION REQUEST CONFIRMATION PANEL
COMMAND ==>

TO CONFIRM CREATION OF:

SOURCE CONFIGURATION FN FT:
FM:   DIRID:

SPECIFY THE FOLLOWING:
PERFORM CREATION          ==> N          (Y or N)

USE ENTER TO PERFORM CREATION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 25. Source Configuration File Creation Request Confirmation Panel

- Select Option 4 (ALTER a management class) and specify a name of an existing management class in the management class name field on the Management Class Application Selection panel to invoke the Management Class Alter panel.

- After choosing the list option on the Management Class Application Selection panel, the Management Class List panel appears. Enter the ALTER line operator next to the management class you want to alter and the Management Class Alter panel is displayed.

The Management Class Define and Alter panels are shown in [Figure 26 on page 49](#) and [Figure 27 on page 50](#).

A blank entered for an attribute value indicates to ISMF that the value has not been specified. A management class cannot be saved if it contains invalid data, or the EXPIRE AFTER DAYS NON_USAGE, EXPIRE SINCE CREATION, or SPACE MANAGEMENT TECHNIQUE fields are left blank.

The Management Class Define panels are primed with the default attribute values before they are displayed. The Management Class Alter panels are primed with the current values of the management class definition before they are displayed.

```
DGTDCMC1          MANAGEMENT CLASS DEFINE PANEL          Page 1 of 2
COMMAND ===>

SOURCE CONFIGURATION FN FT: NEW    CONFIG
FM: A DIRID: VMSYSU:VMTEST0.

MANAGEMENT CLASS NAME: MIGTEST

TO DEFINE MANAGEMENT CLASS, SPECIFY:

DESCRIPTION ===>
===>

EXPIRATION ATTRIBUTES
EXPIRE AFTER NON-USAGE      ===> NOLIMIT      (1 to 9999 or NOLIMIT)
EXPIRE SINCE CREATION       ===> NOLIMIT      (1 to 9999 or NOLIMIT)
EXPIRATION DISPOSITION     ===> ALL          (DATAONLY or ALL)

USE ENTER TO PERFORM VERIFICATION; USE DOWN COMMAND TO VIEW NEXT PANEL;
USE HELP COMMAND FOR HELP; USE END COMMAND TO SAVE AND EXIT; CANCEL TO EXIT.
```

```
DGTDCMC2          MANAGEMENT CLASS DEFINE PANEL          Page 2 of 2
COMMAND ===>

SOURCE CONFIGURATION FN FT: NEW    CONFIG
FM: A DIRID: VMSYSU:VMTEST0.

MANAGEMENT CLASS NAME: MIGTEST

TO DEFINE MANAGEMENT CLASS, SPECIFY:

MIGRATION ATTRIBUTES

PRIMARY STORAGE NON-USAGE   ===> 2          (0 to 9999, NOLIMIT or blank)
SECONDARY STORAGE NON-USAGE ===> 5          (0 to 9999, NOLIMIT or blank)
SPACE MANAGEMENT TECHNIQUE  ===> MIGRATE     (MIGRATE or NONE)

USE ENTER TO PERFORM VERIFICATION; USE UP COMMAND TO VIEW PREVIOUS PANEL;
USE HELP COMMAND FOR HELP; USE END COMMAND TO SAVE AND EXIT; CANCEL TO EXIT.
```

Figure 26. Management Class Define Panels

```

DGTDCMC1          MANAGEMENT CLASS ALTER PANEL          Page 1 of 2
COMMAND ===>

SOURCE CONFIGURATION FN FT: ENGINEER EMP
FM: A DIRID: VMSYSU:VMTEST0.

MANAGEMENT CLASS NAME: PAYROLL

TO ALTER MANAGEMENT CLASS, SPECIFY:

DESCRIPTION ===>
===>

EXPIRATION ATTRIBUTES
EXPIRE AFTER NON-USAGE      ===> 180          (1 to 9999 or NOLIMIT)
EXPIRE SINCE CREATION       ===> 365          (1 to 9999 or NOLIMIT)
EXPIRATION DISPOSITION      ===> DATAONLY    (DATAONLY or ALL)

USE ENTER TO PERFORM VERIFICATION; USE DOWN COMMAND TO VIEW NEXT PANEL;
USE HELP COMMAND FOR HELP; USE END COMMAND TO SAVE AND EXIT; CANCEL TO EXIT.

```

```

DGTDCMC2          MANAGEMENT CLASS ALTER PANEL          Page 2 of 2
COMMAND ===>

SOURCE CONFIGURATION FN FT: ENGINEER EMP
FM: A DIRID: VMSYSU:VMTEST0.

MANAGEMENT CLASS NAME: PAYROLL

TO ALTER MANAGEMENT CLASS, SPECIFY:

MIGRATION ATTRIBUTES
PRIMARY STORAGE NON-USAGE   ===> 2          (0 to 9999, NOLIMIT or blank)
SECONDARY STORAGE NON-USAGE ===> 5          (0 to 9999, NOLIMIT or blank)
SPACE MANAGEMENT TECHNIQUE  ===> MIGRATE    (MIGRATE or NONE)

USE ENTER TO PERFORM VERIFICATION; USE UP COMMAND TO VIEW PREVIOUS PANEL;
USE HELP COMMAND FOR HELP; USE END COMMAND TO SAVE AND EXIT; CANCEL TO EXIT.

```

Figure 27. Management Class Alter Panels

The following is a description of the panel fields displayed in the Management Class Define and Alter panels:

- CONFIGURATION FN FT and FM/DIRID: These fields identify the source configuration file containing the management class being defined or altered.
- MANAGEMENT CLASS NAME: This field contains the name of the management class being defined or altered.
- DESCRIPTION: This field provides you with the opportunity to attach a description to the management class.
- EXPIRE AFTER NON-USAGE (3): This field indicates how many days must elapse since the last file access until the file expires. A NOLIMIT value in this field indicates that the amount of time elapsed since the file was accessed is not used in determining whether the file expires.
- EXPIRE SINCE CREATION (4): This field indicates the minimum number of days after file creation until the file expires. A NOLIMIT value in this field indicates the amount of time elapsed since the file was created is not used in determining whether the file expires.
- EXPIRATION DISPOSITION: This field indicates that either the file data is erased leaving the file structure intact or the file data and file structure are erased. File structure includes existing file authorizations and aliases.

- **PRIMARY STORAGE NON-USAGE:** This field indicates the minimum number of days files associated with the displayed management class must remain unaccessed in primary storage before they are eligible to migrate from primary storage to secondary storage. This eligibility is only checked during DFSMS MANAGE command processing. Specifying NOLIMIT restricts the file to only migration by command.
- **SECONDARY STORAGE NON-USAGE:** This field indicates three items about files associated with the displayed management class: when the value is 0, the file is migrated directly to ML2; if the value is 1–9999 the file is moved to ML2 after spending the specified number of days in ML1; and if the value is NOLIMIT, the file does not migrate or move to ML2.
- **SPACE MANAGEMENT TECHNIQUE:** This field indicates whether files associated with the displayed management class are eligible for migration. The value of NONE indicates that migration is not allowed and MIGRATE indicates that migration is allowed.

Chapter 5. Configurations

Your policies for managing the DASD space of your installation are implemented by defining and activating a source configuration. When a source configuration is activated, it becomes the "ACTIVE" configuration. The ISMF Configuration Application is used for creating, displaying, altering, validating, and activating a source configuration. It is also used for displaying the contents of an active configuration. You can also use the DFSMS ACTIVATE command for activating a source configuration. Configuration files are located in the Shared File System. Additional information about the combinations of the configuration with management classes and ACS can be found in [Appendix H, "Implementing Configurations,"](#) on page 249.

Configuration Application

A configuration, be it a source configuration or an active configuration, contains some basic information associated with it. This information is called the "base configuration information" and contains the following:

- An optional text description of the configuration.
- The name of the default management class associated with this configuration.
- The status of the configuration (valid or invalid).
- An option to invoke, defer, or inherit (disable) ACS processing during file creation. The disable option allows files to inherit a management class by assigning the NULL management class during file creation without invoking ACS. The disable option is indicated by the *INHERIT* value. The defer option indicates that ACS is not invoked when files are created and no management class is assigned to the files being created. The defer option provides the ability to wait until a valid file size is associated with the file. At this time a DFSMS MANAGE, MIGRATE, or CONVERT command can be used to assign a management class.

Note: ACS processing impacts performance since it is additional processing. When the YES option is specified ACS processing occurs for every file creation. The performance impact of the DEFER option is limited to the time when a DFSMS CONVERT, MANAGE, or MIGRATE command is issued. The DEFER option allows you the ability to control when performance is impacted. Overall the INHERIT option has the lowest performance impact, the NULL management class is assigned, but ACS processing is not invoked.

In order to work with the configurations you use the ISMF Configuration Application. The ISMF Configuration Application (option 8 on the ISMF Primary Option menu for storage administrators) offers various facilities to the storage administrator. The Configuration Application:

- Displays the contents of the base configuration information for a source or active configuration
- Defines the base configuration information associated with a source configuration
- Alters the base configuration information associated with a source configuration
- Validates a source configuration
- Activates a source configuration

You use ISMF to create a source configuration and DFSMS/VM stores this in an SFS file. When storage administrators want to display, modify, validate, or activate a source configuration, they must refer to that source configuration by the SFS file ID, for example:

```
srcname1 weekdays poolid1:sms1.sourceconf.
```

The status of a source configuration can be either *valid* (the configuration can be activated) or *invalid* (the configuration cannot be activated). Refer to ["Valid and Invalid Configurations" on page 10](#) for the criteria for a valid configuration. A source configuration is validated through ISMF VALIDATE processing. In addition, whenever a source configuration is saved, the source configuration is marked either valid

or invalid at that time. A valid source configuration is activated by selecting the ACTIVATE option of the Configuration Application Selection panel, issuing the ACTIVATE command on the same panel, or issuing the DFSMS ACTIVATE command.

When a source configuration is activated, DFSMS/VM reads the source configuration file and creates the active configuration. The source configuration, however, is not modified in any manner, and you may display, modify, or even erase that source configuration without affecting the active configuration. This provides you with the ability to keep any number of source configurations, modify them as needed, and activate one at the appropriate time.

During the activation process, a copy of the active configuration is stored into the SFS directory VMSYS:DFSMS.ACTIVECONFIG under the file name and file type that is specified for the keyword ACTIVE_CONFIG_FILE_ID in the DFSMS/VM control file. If DFSMS/VM is shutdown and restarted, it uses the active configuration copy during reinitialization. Since there is a copy of the active configuration, you can modify the source configuration immediately following the activation process without affecting the active configuration.

The only configuration operation that you can invoke on the active configuration is to display the base configuration information. To display the management classes defined in the active configuration, use the Management Class Application (option 3 on the ISMF Primary Option menu). If the configuration contains an ACS routine you can display information, such as, the ACS routine type, last date and time translated, and other descriptive information using the ACS Application (option 7 on the ISMF Primary Option menu).

When a new configuration is activated, all subsequent DFSMS commands that require configuration information use the new configuration, but any DFSMS commands active previous to the activation use the old active configuration. For example, if a DFSMS MANAGE command is issued and processing is not complete when a source configuration is activated, that DFSMS MANAGE command continues using the old configuration. However, a DFSMS MANAGE command issued after the activation uses the new configuration.

Configuration Application Selection Panel

Selection of option 8 (Configuration) on the ISMF Primary Option menu for storage administrators (see [Figure 7 on page 24](#)) results in the invocation of the ISMF Configuration Application Selection panel (see [Figure 28 on page 54](#)).

```
DGTSBSA1          CONFIGURATION APPLICATION SELECTION PANEL
COMMAND ===>

TO PERFORM CONFIGURATION OPERATIONS, SPECIFY:

CONFIGURATION FN FT ===>          (File name and type or 'ACTIVE')
FM/DIRID ===>
===>
===>          (File mode letter or directory name)

SELECT ONE OF THE FOLLOWING OPTIONS ===>

1  DISPLAY - Display base configuration information
2  DEFINE  - Define a base configuration
3  ALTER   - Alter a base configuration
4  VALIDATE - Validate a source configuration
5  ACTIVATE - Activate a configuration

USE ENTER TO PERFORM SELECTION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 28. Configuration Application Selection Panel

The following are descriptions of the panel fields displayed in the Configuration Application Selection panel:

- CONFIGURATION FN FT: This field identifies the configuration to be processed. This field can contain either a CMS file name followed by a CMS file type (when referring to a source configuration) or 'ACTIVE' which performs actions on the active configuration. 'ACTIVE' must be specified with single quotes.

- FM/DIRID: If 'ACTIVE' is specified in the CONFIGURATION FN FT field, this field is ignored. Otherwise this field is used to indicate the directory identifier (dirid) of the source configuration to be processed. Any supported CMS form of *dirid* or file mode is accepted. Refer to the *z/VM: CMS Commands and Utilities Reference* for information about *dirid*. The file mode is a letter assigned to the directory when it is accessed with the ACCESS command. If a file mode is specified it must reference an SFS directory.
- SELECT ONE OF THE FOLLOWING OPTIONS: This field identifies what function the storage administrator wants to perform. Valid values are 1–5.

Configuration Base Display

Option 1 (DISPLAY) displays the base configuration information associated with a source configuration or the active configuration.

The Base Configuration Display panel is shown in [Figure 29 on page 55](#).

```
DGTBSA1          BASE CONFIGURATION DISPLAY PANEL
COMMAND ===>

CONFIGURATION FN FT:
FM:   DIRID:

CONFIGURATION STATUS:
DESCRIPTION:

DEFAULT MANAGEMENT CLASS NAME:
INVOKE ACS FOR FILE CREATION:

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 29. Base Configuration Display Panel

The following are the definitions of the output panels fields:

- CONFIGURATION FN FT: If a source configuration is displayed, these output fields indicate the file name and file type of the source configuration file. If the active configuration is displayed, 'ACTIVE' appears in this field.
- FM: If a source configuration is displayed and a file mode is specified in the FM/DIRID field on the previous panel, the file mode is displayed in this field. If the active configuration is displayed, this field is blank. If a directory ID is specified on the previous panel, this field is blank.
- DIRID: If a source configuration is displayed, the fully qualified directory ID of the source configuration is displayed here. If the active configuration is displayed, this field is blank.
- CONFIGURATION STATUS: This field is VALID or INVALID depending on whether the configuration can be activated (that is, the configuration has been validated and verified successfully). Refer to “[Valid and Invalid Configurations](#)” on page 10 for the criteria for a valid configuration. If a configuration is invalid, run the VALIDATE utility (option 4 on the Configuration Application Selection panel) to obtain a report indicating the reasons for the validation failure.
- DESCRIPTION: If the configuration has a description, it is displayed.
- DEFAULT MANAGEMENT CLASS NAME: This field contains the default management class name. If a file has the NULL management class and the file’s parent directory has the NULL management class, the system default management class is used when processing that file.
- INVOKE ACS FOR FILE CREATION: This field indicates whether to invoke, defer, or disable (inherit) ACS processing during file creation. Values that appear in this field are YES, DEFER, or INHERIT, where:

YES indicates that ACS processing is invoked during file creation. YES should not be used if an installation wants to assign a management class based on file size.

DEFER indicates that ACS processing is not invoked during file creation and no management class is assigned to the new files. The management class is assigned later during DFSMS CONVERT, MANAGE, or MIGRATE command processing

INHERIT indicates that ACS processing is not invoked during file creation and the NULL management class is assigned to the new files. The NULL management class allows a file to inherit the management class of the file's parent directory. If the parent directory management class is also NULL, the system default management class is inherited.

Press END from this panel to return to the Configuration Application Selection panel.

Base Configuration Define or Alter

The DEFINE and ALTER options of the Configuration Application Selection panel are quite similar and are discussed together. They both use as input an SFS file ID specified in the CONFIGURATION FN FT and the FM/DIRID fields located at the top of the panel. DEFINE and ALTER cannot be invoked on the 'ACTIVE' configuration. DEFINE establishes the base configuration information associated with a source configuration, and ALTER modifies existing base configuration information.

When defining the base configuration information, the Source Configuration File Creation Request Confirmation panel shown in [Figure 30 on page 56](#) is displayed when a file does not exist. After inputting the requested information, the Base Configuration Define panel shown in [Figure 31 on page 57](#) is displayed. When you ALTER a source configuration, the Base Configuration Alter panel shown in [Figure 31 on page 57](#) is displayed.

```
DGTDIUX1  SOURCE CONFIGURATION FILE CREATION REQUEST CONFIRMATION PANEL
COMMAND ===>

TO CONFIRM CREATION OF:

SOURCE CONFIGURATION FN FT:
FM:   DIRID:

SPECIFY THE FOLLOWING:
PERFORM CREATION          ===> N          (Y or N)

USE ENTER TO PERFORM CREATION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 30. Source Configuration File Creation Request Confirmation Panel

The Source Configuration File Creation Request Confirmation panel can be displayed from the Configuration Application Selection, Management Class Application Selection, or Translate ACS Routine panels. The Source Configuration File Creation Request Confirmation panel indicates the file ID of the file containing the source configuration to be created. The panel also prompts the storage administrator to either proceed with the creation of the file (by entering Y in the PERFORM CREATION field), or to not proceed (by leaving the N in the PERFORM CREATION field).

The following are descriptions of the panel fields displayed in the Source Configuration File Creation Request Confirmation panel:

- SOURCE CONFIGURATION FN FT: This field identifies the file name and file type of the source configuration file.
- FM: If a file mode is specified in the FM/DIRID field on the previous panel, it is displayed in this field.
- DIRID: The fully qualified directory ID of the source configuration.

- **PERFORM CREATION:** This field is set to N when the panel is displayed. The storage administrator must specifically change this to Y and press ENTER in order to create the file. The valid values are:
 Y—indicating that the source configuration file is created
 N—indicating that the source configuration file is not created

If you indicate to create a file, the Base Configuration Define panel is displayed. The Base Configuration DEFINE or ALTER panels are identical in appearance except for the title and instruction line which becomes DEFINE or ALTER depending on the desired mode. For example:

- When in DEFINE mode, the title is "Base Configuration Define Panel" and the instruction line is "To Define The Base Configuration, Specify"
- When in ALTER mode, the title is "Base Configuration Alter Panel" and the instruction line is "To Alter The Base Configuration, Specify"

The panels are shown in [Figure 31 on page 57](#).

```

BASE CONFIGURATION ALTER PANEL

BASE CONFIGURATION DEFINE PANEL

DGTDBSA1
COMMAND ===>

CONFIGURATION FN FT:
  FM:  DIRID:

CONFIGURATION STATUS:

TO DEFINE THE BASE CONFIGURATION, SPECIFY:
  DESCRIPTION ===>
    =====>

DEFAULT MANAGEMENT CLASS NAME ===>          (1 to 8 characters)
INVOKE ACS FOR FILE CREATION   == => INHERIT  (YES, DEFER, INHERIT)

USE ENTER TO PERFORM VERIFICATION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO SAVE AND EXIT; CANCEL TO EXIT
  
```

Figure 31. Base Configuration Define or Alter Panels

The Base Configuration Define panel is primed with the default attributes values, and the Base Configuration Alter panel is primed with the attribute values previously saved with the base configuration information.

The following describes the panel fields displayed in both the Base Configuration Define and Alter panels:

- **CONFIGURATION FN FT:** This field identifies the file name and file type of source configuration file.
- **FM:** If a file mode is specified in the FM/DIRID field on the previous panel, it is displayed in this field.
- **DIRID:** The fully qualified directory ID of the source configuration is displayed here.
- **CONFIGURATION STATUS:** This field is either VALID or INVALID depending on whether the configuration can be activated. Refer to [“Valid and Invalid Configurations” on page 10](#) for the criteria for a valid configuration. If a configuration is invalid, use option 4 on the Configuration Application Selection panel to obtain a report of why the validation has failed.

Note: If you are defining a base configuration the status is invalid. If the base configuration is the only thing missing, the configuration is valid after the base is defined.

- **DESCRIPTION:** If you want to perform a DEFINE operation, this field is blank, and you can type in up to 120 characters of description information. If you want to perform an ALTER operation, you can edit this

field. This is not a required field, but it is recommended that you provide some descriptive information here.

- **DEFAULT MANAGEMENT CLASS NAME:** This field contains the default management class name. If a file has the NULL management class and the file's parent directory has the NULL management class, the system default management class is used when processing that file.
- **INVOKE ACS FOR FILE CREATION:** This field indicates whether to invoke, defer, or disable (inherit) ACS processing during file creation. Valid values are YES, DEFER, or INHERIT, where:

YES indicates that ACS processing is invoked during file creation. YES should not be used if an installation wants to assign a management class based on file size.

DEFER indicates that ACS processing is not invoked during file creation and no management class is assigned to the new files. The management class is assigned later during DFSMS CONVERT, MANAGE, or MIGRATE command processing

INHERIT indicates that ACS processing is not invoked during file creation and the NULL management class is assigned to the new files. The NULL management class allows a file to inherit the management class of the file's parent directory. If the parent directory management class is also NULL, the system default management class is inherited.

Pressing ENTER verifies the values. If there are syntax errors on any of the values, you are prompted to correct those values. If you have left a required field blank, you are also prompted to provide a value. To save the base configuration information into the source configuration and return to the Configuration Application Selection panel, press END; however, you are not able to save the base configuration information if there are any syntax errors. Enter CANCEL from the ISMF command line to quit without saving.

Validating the Source Configuration

The entire source configuration file or the ACS routine can be validated from the Configuration Application Selection panel by selecting option 4 or entering the VALIDATE command on the command line.

Note: A source configuration or the ACS routine can also be validated from the ACS Application Selection panel.

Before validating the entire source configuration or the ACS routine, the Source Configuration File Validation panel is displayed. The panel is shown in [Figure 32 on page 58](#).

```
DGTDML1          SOURCE CONFIGURATION FILE VALIDATION PANEL
COMMAND ===>

TO PERFORM VALIDATION, SPECIFY:

SOURCE CONFIGURATION FN FT ===>                                (File name and type)
FM/DIRID ===>
===>
===>                                (File mode letter or directory name)

ACS ROUTINE TYPE    ===> *                                (MC=Management Class,
                                                         *=entire Source Configuration File)

LISTING FN FT      ===> DEFAULT LISTING                        (File name and file type or blank)
FM/DIRID ===>
===>
===>                                (File mode letter, directory name or
                                                         blank)

USE ENTER TO PERFORM VALIDATION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 32. Source Configuration File Validation Panel

The following are descriptions of the panel fields displayed in the Source Configuration File Validation panel:

- **CONFIGURATION FN FT:** This field is the fully qualified file name and file type of the configuration to verify.

- FM/DIRID: This field contains the file mode letter or directory ID of the configuration to verify.
- ACS ROUTINE TYPE: This field specifies that either the management class ACS routine is validated or the entire configuration is validated. This field is primed with the last used value.
- LISTING FN FT and FM/DIRID: These fields identify the SFS file containing the results of the validation. The LISTING FN FT field contains the file name and file type. The FM/DIRID field contains the file mode letter or directory ID. If a listing file name and file type are specified in the LISTING FN FT field, the validation results are written to the listing file. If this field is left blank, ISMF provides a default listing. If the file name and file type are entered but the file mode or directory name is not specified, the file is located in the directory accessed as file mode A. If a specified file mode or a defaulted file mode resolves to a minidisk, no verification occurs.

If the listing file ID specified already exists, its contents are replaced with the current output listing. This field is primed with the last listing file ID specified in this panel. When entering this panel for the first time the field is primed with DEFAULT LISTING.

Validation Listing

If a *file ID* is specified in the LISTING FN FT FM field, a validation listing file is created and the Output Listing Disposition panel is displayed. The Output Listing Disposition panel is shown in [Figure 45 on page 74](#). From this panel you can choose to browse the listing file. See [Figure 33 on page 59](#) for an example of browsing the results of a validation.

Note: See the [z/VM: DFSMS/VM Customization](#) for additional information about browse.

```
BROWSE -- LISTING OUTPUT A ----- LINE 000000 COL 001 080
COMMAND ==>                                SCROLL ==> CSR
***** TOP OF DATA *****
                                VALIDATION RESULTS

VALIDATION RESULT:  VALIDATION SUCCESSFUL
SOURCE CONFIGURATION FN FT DIRID: SMALLSYS CONFIG   VMSYSU:VMTEST0.

ACS ROUTINE TYPE:      *
DATE OF VALIDATION:    2001/02/20
TIME OF VALIDATION:    18:43

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

Figure 33. Browsing the Results of an ACS Routine Validation

The Validation Results listing includes the following information:

- VALIDATION RESULT: The result is either VALIDATION SUCCESSFUL or ERRORS DETECTED.
- SOURCE CONFIGURATION FN FT DIRID: This field reflects the source configuration file ID specified on the Source Configuration File Validation panel.
- ACS ROUTINE TYPE: This field reflects the values specified on the Source Configuration File Validation panel.
- DATE OF VALIDATION: The date when the validation occurred. It is in *yyyy/mm/dd* format.
- TIME OF VALIDATION: The time when the validation occurred. The format is *hh:mm* in the range 00:00 to 23:59.

If the validation is not successful, the listing file contains diagnostic messages.

Activating a Configuration

When the definition or modification of a source configuration is complete and validated, the source configuration can be activated. A source configuration file is activated by option 5 on the Configuration Application Selection panel, the ISMF ACTIVATE command, or the DFSMS ACTIVATE command. The

activated source configuration is copied into the active configuration file. The active configuration file is defined in the DFSMS/VM control file by the following statement:

```
ACTIVE_CONFIGURATION_FILE_ID
```

DFSMS/VM automatically uses the configuration contained in the active configuration file at future initializations.

Note: The active configuration file does not contain the ACS REXX and module exits. These routines must be activated as described in: [“Activation of the REXX Exit”](#) on page 192, and [“Activation of the Module Exit”](#) on page 204.

Because DFSMS/VM on one z/VM system is independent of any other DFSMS/VM system on other z/VM systems, each system must activate its own configuration. When the active configuration is changed on one system, the change is not seen by any other system. To have the same active configuration on multiple systems, the configuration must be activated on each system. The source configuration being activated could be the same file, if that file is in a global file pool that is available to other systems. In other words, you could have a source configuration in a single file pool, but you would have to activate this configuration separately on each system requiring the use of a configuration.

Note: If ACS exits (REXX or module) are used, exit files must exist on each system in the SFS directory VMSYS:DFSMS.ACSEXITS at the time of activation.

At the time a source configuration is activated, DFSMS/VM attempts to load the files containing the ACS REXX exit routine and the ACS module exit routine into storage and prepare them for execution. If these files do not exist, a message is displayed on the Activate Request Confirmation panel. An example panel is shown in [Figure 34](#) on [page 60](#). The message varies according to which files do not exist. This message does not represent an error unless you intend for an ACS REXX exit or ACS module exit to be activated. If the message is displayed, you can determine whether to proceed with the activation. You can either enter HELP or press PF1 to display a longer, more explanatory message. Enter HELP or press PF1 again to display a message help panel that provides a detailed explanation of the error along with a suggested action.

Note: The DFSMS ACTIVATE command handles the messages and activation confirmation similarly, see [“DFSMS ACTIVATE”](#) on [page 135](#) for details.

```

                                ACTIVATE REQUEST CONFIRMATION PANEL      EXITS NOT FOUND
COMMAND ===>

  TO CONFIRM ACTIVATION ON THE FOLLOWING CONFIGURATION:

CONFIGURATION FN FT:
  FM:   DIRID:

SPECIFY THE FOLLOWING:
  PERFORM ACTIVATION ===> N      (Y or N)

USE ENTER TO PERFORM ACTIVATION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 34. When No Exits Are Defined

The following are descriptions of the panel fields displayed in the Activate Request Confirmation panel:

- CONFIGURATION FN FT: This field identifies the file name and file type of the source configuration file.
- FM: If a file mode is specified in the FM/DIRID field on the previous panel, it is displayed in this field.
- DIRID: This field contains the fully qualified directory ID as entered on the Configuration Application Selection panel.

- **PERFORM ACTIVATION:** This field is set to N when the panel is displayed. The storage administrator must specifically change this to Y and press ENTER in order to schedule the activation. When activation is completed, a message is issued indicating if the activation is successful. Valid values can be:
 - Y—indicates to activate the configuration
 - N—indicates not to activate the configuration

Chapter 6. Automatic Class Selection

The automatic class selection (ACS) facility assigns management classes to SFS files and directories. The following sections discuss ACS processing and the ACS Application. ACS processing determines the management class for a file or directory. The ACS application allows you to create and alter ACS routines (edit), place the routines in a source configuration (translate), ensure that all management classes assigned by the routine exist in the configuration (validate), and test the ACS routine with simulated input (test). Information about combining ACS processing, the configuration, and management classes is located in [Appendix H, “Implementing Configurations,”](#) on page 249.

ACS Processing

ACS processing assigns a management class name to a file or directory. The actual management class parameters are not stored with the file or directory. This permits the installation to alter management class criteria parameters without running ACS processing against potentially large numbers of files and directories.

There are six conditions that invoke ACS processing on files and directories in a file pool managed by DFSMS/VM:

1. When an SFS directory or subdirectory is created, ACS processing is invoked to assign a management class to the new directory.
2. When a file is recalled from secondary storage, ACS processing is invoked to assign a management class to the recalled file.
3. When a file is migrated via the DFSMS MIGRATE command and the file is not managed by DFSMS/VM (that is, the file has no management class or inherits no management class) ACS processing is invoked on the file. If the file's parent directory has no management class, all directories in the file's path are also processed.
4. When the DFSMS MANAGE command is issued against an SFS storage group and a file or directory has no management class, ACS processing is invoked.
5. When the DFSMS CONVERT command is issued against an SFS storage group and a file or directory has no management class, ACS processing is invoked. Optionally, ACS processing is invoked for all files and directories in the storage group if the DFSMS CONVERT command is invoked with the REDETERMINE option.
6. When an SFS file is created and the active configuration has *invoke ACS for file creation* = YES.

Determining When to Invoke ACS Processing

You determine whether to invoke ACS processing at a directory level or file level. If you choose file level, you then need to determine when to invoke ACS processing. These determinations are made through the *invoke ACS for file creation* field on either the Base Configuration Define or Alter panels, shown in [Figure 31 on page 57](#). There are three values (YES, DEFER, and INHERIT) available for you to choose from:

YES

specifies that ACS processing is invoked during file creation; performance is impacted by this implementation since ACS processing is invoked every time a file is created.

DEFER

specifies that ACS processing is not invoked during file creation and no management class is assigned. ACS processing is invoked later through the DFSMS CONVERT, DFSMS MANAGE, or DFSMS MIGRATE commands.

INHERIT

specifies that ACS processing is not invoked during file creation and the NULL management class is assigned. This is known as directory level processing since file level processing is disabled and

the file *inherits* its parent directories' management class. For additional information about the NULL management class, see [“Rules and Usage of Management Classes” on page 7](#).

High performance is achieved through the INHERIT value (directory level processing); however, the DEFER option provides storage management at the file level and the flexibility to decide when ACS processing takes place (preferably during low system usage hours, thus resulting in minimal impact to system performance). When ACS processing is performed during file creation, the size is unknown. The distinct advantage of deferring ACS processing is that it is performed during DFSMS CONVERT, MANAGE, and MIGRATE command processing when the file size is known. Knowing the file size assists you with determining when and where to migrate files. For example, a system can assign a management class to large files that quickly migrates these files to ML1 or directly to ML2. Knowing the file size, in this case large, determines both when (the file is migrated immediately) and where (the file is migrated to ML1 or directly to ML2).

Customizing ACS Processing

The installation customizes ACS processing according to the requirements of the enterprise. DFSMS/VM provides flexibility to the installation in that it has three ways in which the storage administrator can tailor ACS processing. This is accomplished through any or all of the following:

- ACS routine written by the installation in the ACS language
- ACS REXX installation-wide exit, written by the installation in the REXX language
- ACS module installation-wide exit, written by the installation in either High Level Assembler or Assembler H

The ACS routine is described in more detail throughout this chapter. If you require additional information about the ACS REXX installation exit, see [Appendix A, “ACS REXX Exit,” on page 191](#), or the ACS module installation exit, see [Appendix B, “ACS Module Exit,” on page 203](#).

When ACS processing invokes the ACS routine or the installation-wide exits, it provides various information about the file or directory. This information can be used for deciding what management class should be assigned.

When ACS is processing a *directory*, the following information is passed as read-only variables to the installation ACS routine or installation exit:

- Environment (create, recall, convert)
- Fully qualified directory name
- Existing management class
- Management class of the parent directory
- User ID that is creating the directory or issuing the command

When ACS is processing a *file*, the following information is passed as read-only variables to the installation ACS routine or installation exit:

- Environment (create, recall, convert)
- File name, file type, and fully qualified directory name
- Existing management class
- Management class of the parent directory
- User ID that is creating, migrating or recalling the file or performing DFSMS/VM storage management processing
- Record format of the file
- Size of the file (in kilobytes)

For additional information about the read-only variables, see [“ACS Language Read-Only Variables” on page 179](#).

You can use the values of the read-only variables to determine what the new management class for the file or directory should be and assign a new management class. Or, you can choose to leave the existing

management class there, or even to assign the NULL management class to a file or directory. However, you cannot assign no management class, since no management class means that DFSMS is not managing that file or directory. You can also choose to reject the creation, recall or migration of a file, or the creation of a directory.

ACS Processing Steps

ACS processing for files and directories is performed in the following order:

1. Sets the values of the ACS read-only and read/write variables. The read-only variables are initialized with information about the file or directory being processed. The initial value of the management-class variable (in DFSMS/VM the management-class variable is the only read/write variable) is set according to the following rules:
 - If the file or directory has a management-class name, the management-class variable is initialized with this name.
 - If the file or directory has the NULL management class or no management class, the management-class variable is initialized with the NULL management class.
2. Invokes the ACS routine, if one is provided in the active configuration. Input to the ACS routine includes the read-only variables and the initial management-class value, which can then be modified by the ACS routine.
3. Invokes the REXX exit, if one is active (see [“Activation of the REXX Exit” on page 192](#)). This exit receives the same input values as the ACS routine except that the management-class variable may have been assigned a new value by the ACS routine. The REXX exit can change the management-class variable and thereby override the management-class selection made by the ACS routine.
4. Invokes the module exit, if one is active (see [“Activation of the Module Exit” on page 204](#)). This exit has access to the read-only variables and the latest value of the management-class variable. The module exit can further modify the value of the management-class variable.
5. Validates the value contained in the management-class variable. If the management class is defined in the active configuration, it is considered valid, and is then assigned to the file or directory. Otherwise, the management class is invalid, and the following occurs:
 - If ACS processing is due to a recall, the file is recalled and no management class is assigned to the file.
 - If ACS processing is due to file or directory creation, the file or directory is created and no management class is assigned to the file or directory.
 - If ACS processing is due to DFSMS CONVERT or MANAGE processing, the management class of the file or directory does not change. The file is not migrated or erased.
 - If ACS processing is due to DFSMS MIGRATE processing, the migrate process fails.

If the ACS routine or an exit fails or rejects ACS processing, ACS processing stops (that is, other exits do not execute and validating of the management class is not performed).

Note: If there are no ACS routines or exits (REXX or MODULE), the NULL management class is assigned to files with no existing management class.

The following describes two situations that can occur during DFSMS/VM processing:

- If the ACS routine or exit rejects ACS processing, then create, recall, and migrate fail (the file is not migrated or recalled, or the file or directory is not created). If a DFSMS CONVERT or MANAGE command is issued, no changes are made to the management class of the file or directory currently processing.
- When an active configuration is not available or ACS processing fails, no management class is assigned for create and recall, the management class does not change for convert processing, and migrate processing fails.

Automatic Class Selection Application

The ACS application is invoked when the ACS option is selected on the ISMF Primary Option menu for a storage administrator (see [Figure 7](#) on page 24).

Using ISMF for Automatic Class Selection Tasks

Choose the ACS application when you want to:

- Edit ACS source routines
- Create or replace an existing ACS object table in a source configuration file
- Validate an ACS routine
- Test an ACS routine
- Display information concerning the ACS objects
- Delete an ACS object from a source configuration

Note: An ACS object is an ACS routine that is successfully translated and stored in a configuration file.

The ACS Application Selection panel is shown in [Figure 35](#) on page 66.

```
DGTSFFL1          ACS APPLICATION SELECTION PANEL
COMMAND ===>

SELECT ONE OF THE FOLLOWING OPTIONS ===>

  1  EDIT           - Edit ACS Routine source code
  2  TRANSLATE      - Translate ACS Routines to ACS Object Form
  3  VALIDATE       - Validate ACS Routines Against Storage Constructs
  4  TEST           - Define/Alter Test Cases and Test ACS Routines
  5  DISPLAY        - Display ACS Object Information
  6  DELETE         - Delete an ACS Object from a Source Configuration File

IF OPTION 5 CHOSEN ABOVE, SPECIFY:

CONFIGURATION FN FT ===>          (File name and type or 'ACTIVE')
FM/DIRID ===>
=====
=====          (File mode letter or directory name)

USE ENTER TO PERFORM SELECTION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 35. ACS Application Selection Panel

The following are descriptions of the panel fields displayed in the ACS Application Selection panel:

- **SELECT ONE OF THE FOLLOWING OPTIONS.** This field identifies what function the storage administrator wants to perform. It is always primed with a blank. The options are:
 - Edit ACS routine source code. See [“Edit ACS Routines”](#) on page 67 for more information when option 1 is selected.
 - Translate ACS routines to ACS object form. See [“Translate ACS Routines”](#) on page 67 for more information when option 2 is selected.
 - Validate the ACS routine (or an entire source configuration file). See [“Validating the Source Configuration”](#) on page 58 for more information when option 3 is selected.
 - Define/Alter test cases and Test ACS routines. See [“ACS Test Selection Panel”](#) on page 68 for more information when option 4 is selected.
 - Display ACS object information. See [“Display ACS Object Information”](#) on page 72 for more information when option 5 is selected.
 - Delete an ACS object from a source configuration file. See [“Deleting an ACS Object from a Source Configuration File”](#) on page 73 for more information when option 6 is selected.

- CONFIGURATION FN FT and FM/DIRID. When choosing option 5 these fields identify the configuration file to be displayed. The fields are not used with the other options. They are primed with the file ID of the configuration which was most recently referenced by the user while in ISMF. If the primed value is a source configuration, the fully qualified file name, file type and directory ID are displayed, otherwise, ACTIVE is displayed.

Edit ACS Routines

Selection of option 1 (EDIT) on the ACS Application Selection panel allows the storage administrator to create or modify source code for an ACS routine. When this option is selected, the ACS Source Edit panel is displayed, as shown in [Figure 36 on page 67](#).

```
DGTDFLE          ACS SOURCE EDIT PANEL
COMMAND ===>

TO PERFORM ACS SOURCE EDIT SPECIFY:

ACS SOURCE FN FT ===>          (File name and type)
FM/DIRID ===>
===>
===>          (File mode letter or directory name)

EDITOR NAME      ===>          (PDF or XEDIT)

USE ENTER TO PERFORM EDIT;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 36. ACS Source Edit Panel

The following are descriptions of the panel fields displayed in the ACS Source Edit panel:

- ACS SOURCE FN FT and FM/DIRID. These fields identify the file which contains the ACS routine to be edited. The first time this panel is used these fields are blank. From then on the last ACS source file ID specified on this panel appear in these fields. This source file must be an SFS file, and all forms of the file mode can be used. For additional information about SFS naming conventions, refer to the [z/VM: CMS Commands and Utilities Reference](#).
- EDITOR NAME. This field identifies the editing tool (PDF or XEDIT) for working with the file containing the ACS source routine. It is primed with the last value used or blank if it is being used for the first time.

Translate ACS Routines

The ACS Routine Translation panel, shown in [Figure 37 on page 68](#), is used to translate an ACS routine source file into executable form. The following activities take place during the translation process:

- Check for syntactic and semantic errors.
- Generate an object table if no errors exist.
- Create (if needed) the source configuration file specified in the source configuration file ID input field.
- Place the object table into the source configuration file specified in the source configuration file ID input field. If an object table already exists in the source configuration file, the new object table replaces the existing object table.
- Create a listing file if a file ID is specified in the LISTING FN FT FM field.

```

DGTDFL4          ACS ROUTINE TRANSLATION PANEL
COMMAND ===>

TO PERFORM ACS TRANSLATION, SPECIFY:

SOURCE CONFIGURATION FN FT ===>          (File name and file type)
FM/DIRID ===>
=====
=====          (File mode letter or directory name)

ACS SOURCE FN FT ===>          (File name and file type)
FM/DIRID ===>
=====
=====          (File mode letter or directory name)

LISTING FN FT    ===>          (File name and file type or blank)
FM/DIRID ===>
=====
=====          (File mode letter, directory name or
                    blank)

USE ENTER TO PERFORM ACS TRANSLATION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 37. ACS Routine Translation Panel

The following are descriptions of the panel fields displayed in the ACS Routine Translation panel:

- SOURCE CONFIGURATION FN FT and FM/DIRID. These fields identify the Source Configuration File into which the translated ACS source is placed upon successful translation completion. They are primed with the last source configuration file ID specified on this panel or blank the first time this panel is used. The fully qualified file name, file type, and directory name are displayed.
- ACS SOURCE FN FT and FM/DIRID. These fields identify the file which contains the ACS routine to be translated. They are primed with the last ACS source file ID specified on this panel or are blank the first time this panel is used.
- LISTING FN FT and FM/DIRID. These fields identify the file to contain the output listing from the ACS translator. This file must be an SFS file, and all forms of the file mode are valid. For additional information about SFS naming conventions, refer to the [z/VM: CMS Commands and Utilities Reference](#). These two fields are optional.

ACS Translator Listing

If a file ID was specified in the LISTING FN FT FM field, the listing file is created and the Output Listing Disposition panel is displayed. From this panel you can choose to browse, print, or delete the listing file. The layout of the Output Listing Disposition panel is shown in [Figure 45 on page 74](#). If you choose to browse the listing file, see [Figure 38 on page 68](#) for a sample of the listing file header.

```

ACS TRANSLATOR ***** TIME 16:12:28 DATE 08/28/2001 PAGE 0001 *****
SOURCE CONFIGURATION FN FT DIRID:
ACS SOURCE FN FT DIRID:

```

Figure 38. Header for ACS Translator Listing

ACS Test Selection Panel

Selection of option 4 (TEST) from the ACS Application Selection panel provides the ability to define test cases that simulate the input to ACS for creation, conversion, and recall of files and directories. It also provides the ability to see the results of running the ACS routine with the simulated input. The management class assigned, exit code, and messages written using the WRITE statement are displayed in the listing. When this option is selected, the ACS Test Selection panel is displayed, as shown in [Figure 39 on page 69](#).

```

DGTSFFL2                      ACS TEST SELECTION PANEL
COMMAND ===>

SELECT ONE OF THE FOLLOWING OPTIONS ===>

  1  DEFINE          - Define an ACS Test Case
  2  ALTER           - Alter an ACS Test Case
  3  TEST            - Test ACS Routines

IF OPTION 1 OR 2 CHOSEN ABOVE, SPECIFY:

ACS TEST FN FT ===>                      (File name and type)
FM/DIRID ===>
=====
=====                                (File mode letter or directory name)

USE ENTER TO PERFORM SELECTION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 39. ACS Test Selection Panel

Note: The ACS Test Selection panel is for testing only the ACS routines and excludes both the ACS REXX exit and the ACS module exit.

The following are descriptions of the panel fields displayed in the ACS Test Selection panel:

- SELECT ONE OF THE FOLLOWING OPTIONS. This field identifies the function that the storage administrator wants to perform. The possible options are:
 - When option 1 (DEFINE) is selected, the ACS Test Case Define panel is displayed. This panel is shown in [Figure 40 on page 70](#). The fields ACS Test FN FT and FM/DIRID must be entered.
 - When option 2 (ALTER) is selected, the ACS Test Case Alter panel is displayed. This panel is shown in [Figure 40 on page 70](#). The fields ACS Test FN FT and FM/DIRID must be entered.
 - When option 3 (TEST) is selected, the Test ACS Routines panel is displayed. This panel is shown in [Figure 41 on page 71](#). The fields ACS Test FN FT and FM/DIRID are ignored for this option.
- ACS TEST FN FT and FM/DIRID. These fields identify the file that contains the ACS test case to be created or modified. They are primed with the last ACS test file ID specified on this panel or with blank the first time this panel is used.

ACS Test Case Define or ACS Test Case Alter

The ACS Test Case Define (or Alter) panel allows you to create a new test case or modify an existing test case. In the ACS Test Case Define (or Alter) panel, the storage administrator specifies test values corresponding to the ACS variables listed in [“ACS Language Read/Write Variables” on page 178](#).

Selection of option 1 (Define) or option 2 (Alter) on the ACS Test Selection panel (see [Figure 39 on page 69](#)) results in the display of the ACS Test Case Define or Alter panel (see [Figure 40 on page 70](#)).

The ACS Test Case Define and Alter panels are identical in appearance except for the title and instruction line which becomes DEFINE or ALTER depending on the desired mode. For example:

- In Define mode, the title is "ACS Test Case Define" and the instruction line is "To Define ACS Test Case, Specify"
- In Alter mode, the title is "ACS Test Case Alter" and the instruction line is "To Alter ACS Test Case, Specify"

The panels are shown in [Figure 40 on page 70](#).

The ACS Test Case Alter panel is primed with the current values of the ACS test case that is being altered. All the input fields on the ACS Test Case Define panel are primed with blanks.

ACS TEST CASE ALTER PANEL

ACS TEST CASE DEFINE PANEL

DGTDFFL8
COMMAND ===>

ACS TEST FN FT:
FM: DIRID:

TO DEFINE ACS TEST CASE, SPECIFY:

DESCRIPTION ===>

===>

FN	===>	RECFM	===>	ACSENVIR	===>
FT	===>	SIZE (KB)	===>	MGMTCLAS	===>
FPOOLID	===>	OWNER	===>	PARENTMC	===>
USER	===>	DIR	===>		
PATH	===>				
	===>				
	===>				

USE ENTER TO PERFORM VERIFICATION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO SAVE AND EXIT; CANCEL TO EXIT

Figure 40. ACS Test Case Define/Alter Panel for DFSMS/VM

Note: Only specify values for variables consistent with system limitations. For example:

- The first qualifier in PATH and OWNER must be the same.
- FPOOLID is required.
- SIZE and RECFM are only used in conjunction with FN and FT.

Additional information is available through the ISMF help facility.

The following are descriptions of the panel fields displayed in both the ACS Test Case Define and Alter panels:

- ACS TEST FN FT and FM/DIRID. These fields identify the file containing the ACS test case to be created or modified.
- ACSENVIR. This field identifies the environment where the ACS routine is invoked. For example, the environment could be *create* during file or directory creation, *recall* during file recall, or *convert* when ACS is processing a file or directory during DFSMS CONVERT, MANAGE, or MIGRATE command processing.
- DIR. This field identifies the directory you want ACS to process. When this field is blank it is initialized to NULL.
- FN. This field identifies the file name of the file you want ACS to process. When this field is blank it is initialized to NULL.
- FT. This field identifies the file type of the file you want ACS to process. When this field is blank it is initialized to NULL.
- FPOOLID. This field identifies the file pool name where the directory or the file reside that you want ACS to process. Do not follow the file pool ID entry with a colon (:).
- MGMTCLAS. This field identifies the directories' or file's current management class. When this field is blank it is initialized to NULL.

- OWNER. This field identifies the user ID of the directory or file owner that ACS is processing.
- PARENTMC. This field identifies the management class of the directory specified in the PATH field. When this field is blank it is initialized to NULL.
- PATH. This field identifies the directory name, excluding the FPOOLID, where the directory or file is located. When this field is blank it is initialized to NULL.
- RECFM. This field identifies the record format of the file that you want ACS to process. When this field is blank it is initialized to NULL.
- SIZE. This field specifies the size, in kilobytes, of the file to be processed. It is initialized to zero when the field is blank.
- USER. This field contains the user ID of the person requesting the create, recall, or convert action. This action causes the invocation of ACS.

Test ACS Routines

Selection of option 3 (TEST) on the ACS Test Selection panel provides the storage administrator the capability of testing an ACS routine. The panel shown in [Figure 41 on page 71](#) is displayed.

```

DGTDFFLN          ACS ROUTINES TEST PANEL
COMMAND ===>

TO PERFORM ACS TESTING, SPECIFY:

CONFIGURATION FN FT ===>          (File name and type or 'ACTIVE')
FM/DIRID ===>
    ===>
    ===>          (File mode letter or directory name)

ACS TEST FN FT ===>          (Fully specified or partially
                             specified file name and file type
                             or * * for all files in Fm/Dirid)

FM/DIRID ===>
    ===>
    ===>          (File mode letter or directory name)

LISTING FN FT ===>          (File name and file type or blank)
FM/DIRID ===>
    ===>
    ===>          (File mode letter, directory name or
                             blank)

USE ENTER TO PERFORM VERIFICATION AND TESTING;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 41. ACS Routines Test Panel

Panel Fields

- CONFIGURATION FN FT and FM/DIRID. These fields identify the configuration containing the ACS routine to be tested. They are primed with the file ID of the configuration most recently referenced by the user on this panel. If the primed value is a configuration file ID, the fully qualified file name, file type, and directory name are displayed. Otherwise, ACTIVE is displayed.
- ACS TEST FN FT and FM/DIRID. The file ID containing the ACS test case to be executed. The file mode or directory ID identifies the directory containing the test case files to be processed. This required field must be fully specified. The file name and file type field (*FN FT*) can be fully or partially specified or contain pattern-matching characters.
- LISTING FN FT and FM/DIRID. These fields identify the file containing the output listing from testing the ACS routine. These two fields are optional.

ACS Test Listing

If a file ID is specified in the LISTING FN FT FM field, the listing file is created and the Output Listing Disposition panel is displayed, shown in [Figure 45 on page 74](#). From this panel you can choose to browse the listing file.

The header of the output listing (see [Figure 42 on page 72](#)) includes the fully qualified file ID of the source configuration and the fully qualified directory name containing the executed ACS test cases. The

listing file also contains each ACS test case file name, file type, and the management class name assigned for this test case. The Source Configuration file ID and ACS test file ID reflect the values specified on the ACS Routines Test panel.

```

                                ACS TESTING RESULTS
CONFIGURATION FN FT DIRID:
ACS ROUTINE TYPES:
ACS TEST DIRID:

  ACS TEST FN FT      EXIT CODE  RESULTS
  -----

```

Figure 42. Header for ACS Test Listing

Display ACS Object Information

Selection of option 5 (DISPLAY) from the ACS Application Selection panel provides the:

- Name of the file containing the source code that created the ACS object.
- User ID of the requester that translated the ACS routine.
- Date and time the ACS Object was created.

The ACS Object Display panel is shown in [Figure 43 on page 72](#).

```

DGTIFFL5                                ACS OBJECT DISPLAY PANEL
COMMAND ==>

CONFIGURATION FN FT:
FM:    DIRID:

  ACS RTN   ACS SOURCE FILE NAME, FILE TYPE   LAST TRANS   LAST DATE   LAST TIME
  TYPE      AND DIRECTORY NAME                USERID      TRANSLATED  TRANSLATED
  -----
MGMTCLAS WEEK5   ACS       VMSYSU:VMTEST0.  VMTEST0      2001/05/31  08:08

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 43. ACS Object Display Panel

The following are descriptions of the panel fields displayed in the ACS Object Display panel:

- CONFIGURATION FN FT. This output field displays ACTIVE or identifies the file containing the displayed ACS object.
- FM. If a file mode letter is specified in the FM/DIRID field of the ACS Application Selection panel, the file mode letter is displayed in this field.
- DIRID. If ACTIVE is not displayed, the fully qualified directory identifier is displayed in this field.
- ACS RTN TYPE. This field is always *MGMTCLAS*, that is, the management class ACS routine.
- ACS SOURCE FILE NAME, FILE TYPE AND DIRECTORY NAME. This field identifies the file containing the source code from which the ACS object is created. The file ID is wrapped to the next line when it is more than 33 characters in length. If the ACS object does not exist in the source configuration file, the field contains dashes.
- LAST TRANS USERID. If the ACS object exists in the displayed configuration file, this field contains the user ID of the person who last translated the ACS routine.

- LAST DATE TRANSLATED. If the ACS object exists in the displayed configuration file, this field contains the date when the ACS object was created. The format is yyyy/mm/dd.
- LAST TIME TRANSLATED. If the ACS object exists in the displayed configuration file, this field contains the time when the ACS object was created. The format is hh:mm.

Deleting an ACS Object from a Source Configuration File

The ACS object is deleted from a source configuration file by using option 6 (DELETE) from the ACS Application Selection panel. This panel asks you to certify that the ACS object should be deleted. The ACS Object Delete panel is shown in [Figure 44 on page 73](#).

```

DGTDFLI          ACS OBJECT DELETE PANEL
COMMAND ===>

TO DELETE AN ACS OBJECT, SPECIFY:

SOURCE CONFIGURATION FN FT ===>          (File name and type)
FM/DIRID ===>
      ===>          (File mode letter or directory name)
      ===>

ACS ROUTINE TYPE      ===> MC          (MC=Management Class)

USE ENTER TO PERFORM ACS DELETION;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 44. ACS Object Delete Panel

The following are descriptions of the panel fields displayed in the ACS Object Delete panel:

- SOURCE CONFIGURATION FN FT and FM/DIRID. These fields identify the source configuration file where the ACS object is to be deleted. They are primed with the source configuration file ID specified by the user while in ISMF. The fully qualified file name, file type, and directory name are displayed.
- ACS ROUTINE TYPE. This field identifies the type of ACS object. In DFSMS/VM, this field always contains the value MC.

To perform the ACS object deletion you need to press enter.

Important: After deletion the configuration should not be activated unless ACS exits exist, or an ACS routine is added to it through translation. If it is activated, all files and directories with no management class are assigned the NULL management class.

Output Listing Disposition

The Output Listing Disposition panel, shown in [Figure 45 on page 74](#), is displayed when the storage administrator specifies the output listing file ID on the Source Configuration File Validation panel, Translate ACS Routine panel, or the Test ACS Routines panel.

```

DGTDFFLC          OUTPUT LISTING DISPOSITION PANEL
COMMAND ===>

LISTING FN FT:
FM:  DIRID:

SPECIFY OUTPUT LISTING DISPOSITION:

BROWSE OUTPUT LISTING      ===> Y      (Y or N)

PRINT OUTPUT LISTING       ===> N      (Y or N)

DELETE OUTPUT LISTING      ===> N      (Y or N)

USE ENTER TO PERFORM SELECTION; USE CANCEL COMMAND TO CANCEL;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 45. Output Listing Disposition Panel

The following are descriptions of the panel fields displayed in the Output Listing Disposition panel:

- LISTING FN FT and FM/DIRID. These output fields identify the file containing the output listing from testing the ACS routine.
- BROWSE OUTPUT LISTING. If Y (yes) is entered in this field, then the availability of BROWSE utilities is checked. If either ISPF/PDF or CUF BROWSE facilities are available, BROWSE is invoked on the listing file. If neither of the facilities are available a message is issued indicating that the BROWSE function is not available.
- PRINT OUTPUT LISTING. If Y (yes) is entered in this field, the output listing file is printed. This field is required, and it is primed with N.
- DELETE OUTPUT LISTING. If Y (yes) is entered in this field, the listing file is erased after printing, browsing, or exiting from this panel. This field is required, and it is primed with N.

Chapter 7. Specifying DFSMS/VM Secondary Storage

Secondary storage is the space that you specify for the storing of migrated files. Secondary storage consists of migration level 1 (ML1) and migration level 2 (ML2). Data migrated to ML1 is stored on DASD, data migrated to ML2 is stored primarily on tape, but can also be stored on DASD. The `MIGRATION_LEVEL_1` keyword must be specified if ML2 is enabled. The following sections primarily discuss ML1; for more information pertaining to specifying ML2, see [z/VM: DFSMS/VM Customization](#).

Naming a Migration Level 1 File Space

Note: If you are attempting to *rename* ML1, see “Renaming Migration Level 1 and Subdirectory Names” on page 76 for information to assist you with this task.

Although ML1 can be in its own file pool or in a file pool containing other data, IBM recommends that you define a file pool dedicated to only ML1. The file pool and file space used for ML1 is specified by the `MIGRATION_LEVEL_1` control file statement. The file pool specified in the `MIGRATION_LEVEL_1` statement can be a local or a remote file pool. If you choose to use a remote file pool, be aware that a performance degradation may be experienced during migration and recall operations. If the remote file pool cannot be accessed (links down, system down), migrated files are unavailable. Consider placing ML1 in a file pool that has slower DASD than the DASD used for primary storage. Also, be aware that the VMSYS file pool cannot be specified as the ML1 file pool. If VMSYS is specified, DFSMS/VM does not initialize and an error message is issued indicating that an invalid file pool for ML1 has been specified. If ML1 resides on a file pool that contains user data managed by DFSMS/VM, then ML1 data **must** reside in a separate storage group. Also, if you place ML1 data in the same storage group as the work directory or the log files, you cannot backup this storage group, using `FILEPOOL BACKUP` while DFSMS/VM is active.

The file space (that is, top-level directory or user ID) specified by the `MIGRATION_LEVEL_1` control file statement must begin with the characters **DFSMS**, for example, `FPOOLID:DFSMSxxx`. (the ending period is required). The file space name can be from five-to-eight characters long. DFSMS/VM creates subdirectories in this file space, so the file space must be enrolled in this file pool. The DFSMS master and servers also must be storage administrators to this file pool. To determine the amount of space required, see “Determining the Amount of Migration Level 1 Space” on page 76.

The following are examples that specify the ML1 file space:

1. To specify a local file pool:

```
MIGRATION_LEVEL_1 SFS VMSYSU:DFSMS001.
```

2. To specify a remote file pool:

```
MIGRATION_LEVEL_1 SFS FPOOL3:DFSMSX1.
```

You must also specify the `REMOTE` start-up parameter in the `DMSPARMS` file for the particular file pool.

It is an SFS convention that file pool IDs beginning with `VMSYS` are always local. Those without this name prefix can be local or remote. For additional information, refer to the [z/VM: CMS File Pool Planning, Administration, and Operation](#).

Specifying No Secondary Storage

If your installation does not want to migrate files or wants to change from a migration environment to an expiration-only environment, you need to exclude the `MIGRATION_LEVEL_1` and the `MIGRATION_LEVEL_2` statements from the control file. If the statements are not specified, DFSMS/VM only performs expiration of files during DFSMS `MANAGE` command processing. DFSMS/VM recognizes the absence of a secondary storage file space during initialization and fails all migration requests that it receives. The message placed in the report file indicates that migration is not allowed on this system. If

you are changing from a migration to a expiration-only environment, be sure to recall all migrated files before changing to the expiration-only environment (the DFSMS REPORT command generates a report of migrated files). Once these control file statements are excluded, any previously migrated files existing in secondary storage cannot be recalled.

Other ways of changing the environment without recalling migrated files are to replace the installation-wide migration exit to prohibit file migration or to change all management classes to expire files only. This allows you to recall files as needed but prevents file migration.

If there is a need to remove DFSMS/VM, see [z/VM: DFSMS/VM Customization](#) for instructions.

Determining the Amount of Migration Level 1 Space

The amount of DASD space you need for ML1 depends on your installation's data reference patterns and the type of management classes. Use the following formula (see [Figure 46 on page 76](#)) to estimate DASD space requirements:

$$\left[\begin{array}{c} \text{(Number of 4K blocks in all} \\ \text{DFSMS/VM-managed local} \\ \text{and remote file pools)} \end{array} \right] * \left[\begin{array}{c} \text{(Percentage of data} \\ \text{that you expect to} \\ \text{migrate to ML1)} \end{array} \right] * \left[\begin{array}{c} 1 \\ \hline 3 \end{array} \right]$$

Figure 46. Equation to Determine the Migration Level 1 Space

Ensure that the ML1 file space is enrolled with at least this amount of space and that enough physical space is allocated. If a large portion of the files to be migrated are under four 4K blocks, then the number of blocks for ML1 should be increased (the smallest file in secondary storage is one 4K block). File migration to ML1 fails when all allocated ML1 space is used. To eliminate the possibility of a file migration failure, you need to monitor secondary storage space utilization and increase storage when required.

Recommendation: Allocate a logical limit that is the same as the number of physical blocks in the file pool. Set the GROUPTHRESH (file pool startup parameter) limit to at least 95. This way, either the QUERY FILEPOOL STORGRP or the QUERY LIMITS command can be used to monitor storage. Warning messages will be sent to the DFSMS log and to the file pool console when ML1 usage exceeds the GROUPTHRESH limit.

Renaming Migration Level 1 and Subdirectory Names

This section describes naming the ML1 subdirectory, renaming the ML1 file pool and file space, renaming a primary file pool containing migrated data, renaming the fully qualified LU name or global resource ID, and abnormal conditions. Renaming ML1 is not recommended. Therefore, be careful when choosing the name for ML1. However, if changes must be made, the following sections indicate how to modify ML1 to reflect the new names.

Because DFSMS/VM uses the primary file pool (that is, the source file pool) identifier in its ML1 directory structure, renaming a file pool affects the ability of DFSMS/VM to recall files. If an installation wishes to rename a file pool, they must also rename the directory in ML1. Before performing any type of file pool renaming (except primary file pool renaming), you must stop DFSMS/VM. For more information about SFS file pool naming conventions, see the [z/VM: CMS File Pool Planning, Administration, and Operation](#).

Naming the Migration Level 1 Subdirectories

When DFSMS/VM migrates a file from the general user's file space, it places the compacted data from the file in an ML1 subdirectory. The subdirectory names are created from entries in the DFSMS/VM control file. If you change the DFSMS/VM control file parameters used for the ML1 subdirectory names, you must also rename the subdirectories containing ML1. The following provides items that determine the subdirectory name:

```
file pool:fileSPACE.MIGRATIONLEVEL1.netid.luname.global_resource_id.primaryfilepool.SGnn
```

file pool

The SFS file pool containing ML1 data. This is obtained from the MIGRATION_LEVEL_1 statement in the control file.

filespace

The space allocated within a file pool for ML1 data. This is obtained from the MIGRATION_LEVEL_1 statement in the control file.

MIGRATIONLEVEL1

This is a constant value.

netid.luname

netid and *luname* are two subdirectories that are identified by the FULLY_QUALIFIED_LUNAME control file statement. This DFSMS/VM control file statement identifies a unique network addressable unit that is used for routing within a Systems Network Architecture (SNA network). For additional information about the control file statement, see [z/VM: DFSMS/VM Customization](#).

global_resource_id

The global APPC/VM resource ID of the DFSMS/VM subsystem managing the file pool. This is obtained from the GLOBAL_RESOURCE_ID statement in the control file. The GLOBAL_RESOURCE_ID needs to be a unique resource ID over the entire TSAF collection.

primaryfilepool

The identifier of the file pool from which the file is migrating.

SGnn

SG followed by (nn) the storage group number (in decimal) from which the file is migrating.

For example:

```
MIGLEV01:DFSMS.MIGRATIONLEVEL1.USIBM.SUA.DFSMS01.VMSYS1.SG2
```

Renaming the Migration Level 1 File Pool

If you are renaming the ML1 file pool, follow these steps:

1. Stop DFSMS/VM before performing any of the following steps (use the DFSMS STOP command).
2. Rename the file pool. This includes renaming the file pool ID in DMSPARMs and renaming the POOLDEF file.
3. Update the control file with the new ML1 file pool name.
4. Start the virtual machine that starts the renamed file pool.
5. Restart DFSMS/VM.

Renaming a Primary File Pool Containing Migrated Data

If you are renaming a primary file pool that contains migrated files, follow these steps:

Note: You do not need to stop DFSMS/VM to perform this operation unless the DFSMS/VM virtual machines use the file pool being renamed for the work directory, logging directory, or directories in the file pool are referenced by installation-wide exits.

1. Recall all ML2 data for this file pool.
2. Stop the file pool.
3. Rename the file pool; this includes renaming the file pool ID in DMSPARMs and renaming the POOLDEF file.
4. If the migrated data is stored in the primary file pool being renamed, the file pool needs to be restarted.
5. Rename the ML1 directory for the migrated files, see [“Naming the Migration Level 1 Subdirectories” on page 76](#) for a description of the *primary file pool* parameter.
6. Start the renamed file pool.

For example, if you were to rename the file pool ID from *VMSYS1* to *VMXXX1* and *VMSYS1* is a primary file pool which has migrated files, you must rename the subdirectory corresponding to that file pool in the ML1 file space as follows:

```
Original version:
DFSMS.MIGRATIONLEVEL1.USIBM.SUA.DFSMS01.VMSYS1

Renamed version:
DFSMS.MIGRATIONLEVEL1.USIBM.SUA.DFSMS01.VMXXX1
```

Note:

1. If the control file contains a reference to the primary file pool ID, be sure to change the control file reference when renaming the file pool ID. DFSMS/VM must be stopped and restarted to pick up changes to the control file.
2. Do the following if it is not feasible to recall ML2 files, but renaming the primary file pool is necessary:
 - a. Use the CMS FILEPOOL BACKUP command to backup each storage group in the file pool.
 - b. Use the CMS FILEPOOL RESTORE command on each storage group to restore the file pool to the new file pool. See [Chapter 8, “DFSMS/VM Backup and Recovery Processing,” on page 81](#).

Renaming the Fully Qualified LU Name or Global Resource ID

As with renaming a file pool, renaming the fully qualified LU name or the global resource ID in the DFSMS/VM control file necessitates the renaming of qualifiers in the ML1 file space (in the previous example, USIBM.SUA and DFSMS01 respectively). Again, you must rename all changed directories.

Renaming the ML1 File Space

If you are renaming the ML1 file space, follow these steps:

1. Stop DFSMS/VM before performing the following steps (use the DFSMS STOP command).
2. Begin the new file space name with the characters DFSMS and do not locate the file space in any storage group that contains DFSMS/VM managed user data. Rename by issuing the FILEPOOL RENAME command, refer to [z/VM: CMS File Pool Planning, Administration, and Operation](#).
3. Update the DFSMS control file with the new file space name.
4. Restart DFSMS/VM.

Abnormal Conditions

For various reasons, it is possible that the contents of the DFSMS/VM and SFS file pools do not match. One reason for the mismatch could be that SFS performs a function (such as erasing a file) when DFSMS/VM is not active. The following conditions are considered abnormal:

- The file has an SFS file pool catalog entry that does not show the file as migrated, but DFSMS/VM has an entry for the file in secondary storage.

This condition can occur if you erase the migrated file when DFSMS/VM is not active and then recreate a file that has the same internal identifier. This problem is detected with the DFSMS REPORT command. To fix this condition, use the token provided in the DFSMS REPORT command output as the input token to the DFSMS DELETE command.

- The file has an SFS file pool catalog entry that shows the file as migrated, but DFSMS/VM has no entry for the file. This occurs if control file parameters have been changed and the corresponding secondary storage hierarchy has not been modified to reflect these changes. If this is not the case, then recover the files from backup, see [Chapter 8, “DFSMS/VM Backup and Recovery Processing,” on page 81](#).

This condition may also occur if primary or secondary storage is restored after a DASD failure without synchronizing primary and secondary storage. Refer to [Chapter 8, “DFSMS/VM Backup and Recovery Processing,” on page 81](#) for backup and recovery considerations.

- Abnormal conditions may also occur in ML2 if installation defined Tivoli® Storage Management (TSM) policies erased ML2 data. The recommendation is that TSM policies do not erase DFSMS/VM data stored in their repositories but to have DFSMS/VM manage their own data.

Important: If you rename the primary file pool without first recalling data migrated to ML2, you are not be able to recall the data.

Chapter 8. DFSMS/VM Backup and Recovery Processing

The introduction of migration into your system is likely to cause changes in the way you perform backup processing. With migration capability the file data is not necessarily in the primary storage group. As a result, you need to take extra steps to make sure all required data is backed up. This publication assumes that you are using SFS facilities for backing up your SFS file pools. If you are not, be sure to refer to [“Using Non-IBM Backup Facilities”](#) on page 82 and [“Using Non-IBM Facilities for Recovery Processing”](#) on page 84 before devising your recovery procedures. The following commands are available for backup and recovery of files located in SFS storage groups: FILEPOOL BACKUP, FILEPOOL RESTORE, and FILEPOOL FILELOAD.

Important: If migrated data exists and migration level 2 (ML2) is defined in the DFSMS/VM control file then IBM SAA® AD/Cycle® Language Environment/370 (LE/370) must be installed and accessible from the user ID issuing the SFS utility backup and restore commands. The disk containing the LE/370 SCEERUN load library must be linked and accessed prior to issuing the SFS backup and restore commands if any migrated data exists on your system. If LE/370 is not installed on your system the following commands cannot backup or restore migrated data:

FILEPOOL BACKUP
FILEPOOL RESTORE
FILEPOOL FILELOAD

The following sections discuss backup and recovery process within the DFSMS/VM environment. For information about recovery procedures strictly within the SFS and CRR environments, refer to [z/VM: CMS File Pool Planning, Administration, and Operation](#).

Backup Processing

Backup processing is essential to any computing environment. The following sections discuss the IBM SFS FILEPOOL BACKUP utility and non-IBM backup facilities.

Using the SFS FILEPOOL BACKUP Utility

Almost every SFS file should be backed up regularly. The only files that you would not back up are files that you can afford to lose at any time. IBM provides SFS file protection through the SFS FILEPOOL BACKUP command. During SFS storage group backup processing, the associated ML1 data is also backed up. As a result, if you are using the FILEPOOL BACKUP command, when a primary storage group is restored using the SFS FILEPOOL RESTORE command, the associated ML1 data is also restored. That is, your migrated data for that storage group is restored in synchronization with the primary storage group.

Because ML2 data is probably located on tape, SFS FILEPOOL BACKUP does not back up the data associated with ML2 files. Recalling files on ML2 tape for backup is unacceptable from a performance and usability perspective and defeats the value of migration. Instead the SFS FILEPOOL BACKUP command backs up information (called ML2 entries) regarding the ML2 files, thereby allowing recovery from the ML2 tape. The SFS FILEPOOL BACKUP command will create a new backup entry in the Tivoli Storage Management (TSM) (ML2) database for each ML2 entry. This backup entry can be used to reference the ML2 data even after the ML2 entry is deleted (for instance, after an erase or recall). As a result, ML2 data is also recovered when using the SFS storage group or file level restore. This support requires no tape mounts for either backup or recovery.

These SFS backup entries in the ML2 database will continue to exist until deleted through use of the DFSMS DELETE ML2BACKUP command. The DFSMS DELETE ML2BACKUP command should be used when no valid SFS backup tapes (with associated ML2 data) exist prior to a specific date. The DFSMS DELETE ML2BACKUP command takes a date as input and deletes all SFS backup entries in the ML2 database prior to that date. This command does not affect migration entries. The DFSMS DELETE

ML2BACKUP command should be used as previous SFS backup tapes are recycled. Issuing the DELETE ML2BACKUP is the only way to free resources used by TSM.

Use of the SFS backup utility allows you to recover data (migrated or not) in the following situations:

- Accidental erasure of primary files that are migrated to ML1 or ML2
- Accidental erasure of data stored in ML1
- Failure of primary SFS catalog (any or all DASD)
- Failure of primary SFS data (any or all DASD)
- Failure of ML1 SFS catalog (any or all DASD)
- Failure of ML1 SFS data (any or all DASD)

DFSMS/VM stores ML2 data in the TSM server. Protection for ML2 data should be provided by following TSM procedures for backing up TSM data. These procedures include TSM database and log mirroring. In addition, since ML2 data on TSM volumes may be the only version of the migrated data, it is recommended that TSM volumes (including tapes) be copied using any volume copy facility. If TSM data is lost, then follow TSM procedures for restoring data.

The SFS commands restore storage groups to the state of the files at the time of the storage group backup. In addition to restoring data at a storage group level, you can restore at the file level through the SFS FILEPOOL FILELOAD command. Individual files from SFS storage groups are restored to the state of the storage group backup.

You should consider placing the DFSMS/VM log files and work directory in their own storage group (or in a storage group containing data that does not require backup and does not contain secondary storage). The log files and work directory are sometimes open during DFSMS/VM processing. If any files are open in a storage group being backed up, FILEPOOL BACKUP processing fails.

Be careful not to confuse the FILEPOOL BACKUP command with the FILESERV BACKUP command. The FILESERV BACKUP command backs up the control data, not user storage groups. For details about the SFS FILEPOOL BACKUP, RESTORE, and FILELOAD and SFS FILESERV BACKUP commands, refer to [z/VM: CMS File Pool Planning, Administration, and Operation](#).

Using Non-IBM Backup Facilities

If you are considering using a non-IBM facility for backing up and restoring your SFS data, you should determine whether the facility is aware of the interrelationships of SFS data (primary and migrated data). The facility you choose must backup and restore primary entries in synchronization with the associated migrated data to avoid loss of data. Refer to the documentation for the facility to determine how to back up your data.

Other backup facilities, such as the DASD dump restore (DDR) service program, backup physical DASD space without regard for the meaning of the data. That is, there is no regard for the logical relationships of the data within SFS. Because of the difficulties involved with synchronizing physical backups and restores, it is recommended that such physical backup facilities not be used to protect SFS storage groups and their migrated data.

For additional information, see [“Using Non-IBM Facilities for Recovery Processing”](#) on page 84.

ML1 Backup Considerations

In order to successfully recover ML1 data, that data must be backed up in synchronization with primary and restored in synchronization with primary. The facility you choose to perform full-backup processing on a storage group must also backup the associated ML1 directory (again using the non-IBM facility). The ML1 directory name for a storage group is based on statements in the control file, the file pool being backed up, and the storage group being backed up. The structure of the ML1 directory name is shown below.

```
file pool:filespace.MIGRATIONLEVEL1.netid.luname.global_resource_id.primaryfilepool.SGnn
```

file pool

The SFS file pool containing ML1 data. This is obtained from the MIGRATION_LEVEL_1 statement in the control file.

filespace

The space allocated within a file pool for ML1 data. This is obtained from the MIGRATION_LEVEL_1 statement in the control file.

MIGRATIONLEVEL1

This is a constant value.

netid.luname

netid and *luname* are two subdirectories that are identified by the FULLY_QUALIFIED_LUNAME control file statement. This DFSMS/VM control file statement identifies a unique network addressable unit that is used for routing within a Systems Network Architecture (SNA network). For additional information about the control file statement, see [z/VM: DFSMS/VM Customization](#).

global_resource_id

The global APPC/VM resource ID of the DFSMS/VM subsystem managing the file pool. This is obtained from the GLOBAL_RESOURCE_ID statement in the control file. The GLOBAL_RESOURCE_ID needs to be a unique resource ID over the entire TSAF collection.

primaryfilepool

The identifier of the file pool from which the file is migrating.

SGnn

SG followed by (nn) the storage group number (in decimal) from which the file is migrating.

For example:

```
MIGLEV01:DFSMS.MIGRATIONLEVEL1.USIBM.SUA.DFSMS01.VMSYS1.SG2
```

In order to backup the migrated data associated with the primary storage group, you need to backup **all** files in the corresponding ML1 directory. Back up these files at the same time as the primary group and you should ensure that no migrations, recalls or erases occur from the time the storage group is backed up until the associated ML1 directory is backed up.

ML2 Backup Considerations

TSM stores ML2 data in the TSM server. Protection for ML2 data should be provided by following TSM procedures for backing up TSM data. These procedures include TSM database and log mirroring. In addition, since ML2 data on TSM volumes may be the only version of the migrated data, it is recommended that TSM volumes (including tapes) be copied using any volume copy facility. If TSM data is lost, then follow TSM procedures for restoring data.

Recommendations

For the protection of migrated data, we recommend that you do the following, no matter what backup facilities and procedures you follow.

1. When considering backup procedures for your installation, you need to coordinate the criteria used for migration and backup of SFS files. For example, if you do full backups weekly and you want to guarantee that each file is included in at least one full backup before migration, then set the *Primary Days Non-Usage* management class attribute to more than seven days. This provides enough time to back up the file before it is eligible for migration. Ensure that your management classes allow migration of data only after a period longer than the storage group backup interval.
2. The ML1 storage group should be protected through regular backups.
3. If using ML2, the TSM database and log should be protected through mirroring.
4. Since ML2 data on TSM volumes are not backed up after it is migrated, a tape containing ML2 data remains a single point of media failure for data loss. We recommend that you perform tape dual copy as a regular part of TSM (ML2) processing. This ensures that a copy exists for the ML2 data.

5. You may want to use the FSMMECHK CSL exit to guarantee that you have a backup copy of a file before you allow it to be migrated or expired. This is recommended especially if you have ML2 data or are using a non-IBM backup facility. For additional information about using the FSMMECHK CSL exit see, *z/VM: DFSMS/VM Customization*.
6. In your installation's backup and recover plan you should save two or three backup versions in case your most recent backup is lost or destroyed. For example, if you must restore your storage group from a backup tape and the tape breaks, the storage group backup data is lost unless you have saved an older backup.

Recovery Processing

Recovery processing, also known as restoring data, consists of recovering an entire storage group or single files within a storage group. The following sections discuss recovering a storage group and a single file, and recovery when the IBM backup product or when a non-IBM backup product has been used. When recovering any data, you need to inform general users of the impending recovery. General users need to be aware of the recovery because the storage group or file data can regress to an earlier state (depending on whether any changes were made after the last backup). Furthermore, the storage group is not available for read or write access during recovery processing. You should advise the general users against attempting to reference the storage group until recovery processing completes.

Recovering an Entire Storage Group

Recovery of an entire storage group is necessary if, for example, a device in the storage group fails. The recovery procedure differs depending upon whether the SFS FILEPOOL BACKUP command or a non-IBM backup product has been used to backup the storage group. You may want to run a DFSMS REPORT SPACEMANAGEMENT STORGROUP command after recovery to guarantee the integrity of the storage group.

Storage Groups Backed Up with the FILEPOOL BACKUP Command

The SFS FILEPOOL BACKUP command backs up both primary storage and secondary storage. You can issue a FILEPOOL RESTORE command to recover the primary storage group, the ML1 directory, and any ML2 entries to the state of the backup, if the SFS FILEPOOL BACKUP command was used to perform the backup. If you are attempting to recover from TSM ML2 failures, refer to TSM recovery procedures. In this case, no recovery of primary or ML1 is needed.

Using Non-IBM Facilities for Recovery Processing

Non-SFS backup products do not necessarily have the same support of secondary storage files as SFS FILEPOOL BACKUP processing does. As a result, special care is necessary during restore processing to allow for a complete recovery of your storage group.

During restore processing, do not allow migrate, recall or erase processing to take place; this can cause an out-of-sync condition. You must either shutdown DFSMS/VM or lock the storage group until the restore is complete. The primary storage file pool and ML1 file pool must be coordinated when using a non-IBM backup product. Failure to restore primary file entries in synchronization with the migrated data in ML1 may cause data loss.

The first step in the recovery process is to recover the storage group from the last full-backup. This regresses the storage group to the point of your last full-backup. Your non-IBM restore facility must recover the ML1 directory for this storage group to the same point in time as the primary storage group so that primary storage and ML1 are synchronized. Refer to [“ML1 Backup Considerations” on page 82](#) for more information.

Many installations perform some form of incremental backup processing ¹ using either their own solution or one offered by a non-IBM product. If this is true for you, after you recover the storage group, you

¹ *Incremental backup* is the process of backing up **only** those files which have been changed since the last backup was done.

need to recover each of the files backed up during incremental processing. As mentioned in [“Recovering Specific Files”](#) on page 85, most of the time files backed up during incremental processing are not migrated, since they have been changed (and therefore referenced) recently. Because of this, you simply need to perform your incremental backup to the storage group which was just restored. Upon doing so, the storage group is recovered back to the latest incremental processing point.

Recovering Specific Files

If you need to recover a single file, the recovery procedure varies depending upon the backup mechanism you use and whether the file was migrated when it was last backed up.

Using the SFS FILEPOOL FILELOAD Command

If you want to recover one or more files from a backup created by FILEPOOL BACKUP processing, use the FILEPOOL FILELOAD command. Unlike the FILEPOOL RESTORE command, FILEPOOL FILELOAD lets you restore only the portion of the storage group data that you need. The SFS FILEPOOL FILELOAD command restores primary files and data, and any ML1 data or ML2 entries for any files being restored. If you are recovering from loss of ML2 data in TSM, refer to TSM recovery procedures. In this case, no recovery of primary or ML1 is needed. For additional information about using the FILEPOOL FILELOAD command, refer to [z/VM: CMS File Pool Planning, Administration, and Operation](#).

Recovering Files Using a Non-IBM Backup Product

If the file was backed up using a non-IBM backup product, the ability to recover the file depends upon how the backup product handles migrated files. There are two possible ways of handling migrated files:

- Force a recall of the file to back it up
- Perform no backup at all and guarantee, through installation procedures that files are backed up before they are eligible to be migrated

In both instances, the file can be recovered from the previous backup version.

It is possible for a file to be migrated before it ever gets backed up. This can generally be prevented, though, by controlling the backup and migration frequency. Suppose, for example, that you run your backups every other night, and that all of your management classes have at least a seven-day waiting period before a file gets migrated. In this case, the file will be backed up before it is eligible for migration. To determine when the file was migrated, you have two options:

- Examine the log files of the DFSMS server machines for message FSM3209I. This message indicates the files that were successfully migrated. Keep these files for future reference, do not discard them.

Recommendation: Log all messages when using a non-IBM product.

- Examine a recent copy of the output of the DFSMS REPORT SPACEMANAGEMENT FILESP command. If the file was migrated, the report indicates when it was migrated. This examination also requires that you have taken some action previously to generate and save the report on a somewhat frequent basis.

A non-IBM backup product may perform actions other than those mentioned here. If that is the case, you must consult the reference books of that product to determine how to recover the lost file.

Recovering From a Secondary Storage Failure

It is possible for secondary storage (either ML1, ML2, or both) to incur a media failure. If this occurs, when recovering secondary storage, you will need to ensure that primary and secondary storage remain in synchronization.

Recovering From an ML1 Failure

It is possible for files in the ML1 file space to be lost as a result of media failure. If you incur a media failure or some other catastrophic error in your ML1 file space, you need to recover the entire storage group containing ML1 that was damaged.² The procedure for this is outlined in [“Recovering an Entire Storage Group”](#) on page 84, with the exception that the storage group being recovered contains no

migrated files. This regresses the storage group containing ML1 to the time of the last backup. You then need to take manual action to resynchronize ML1 with primary storage. To resynchronize ML1 with primary storage, you must first issue the DFSMS REPORT SPACE MANAGEMENT STORGROUP command for each primary storage group which uses the ML1 storage group.

Note: Multiple storage groups can be involved since migration usually occurs from multiple rather than a single storage group.

Next, examine the report and perform, **when applicable**, the following actions in the listed order:

1. If files are listed that are not migrated but migrated data exists for these files (ML1, ML2 or both), issue the DFSMS DELETE MIGRATION command to delete these orphans. This ML1 situation typically arises when files have been recalled since the last backup (restoring ML1 brings back ML1 data). For more details on the DFSMS DELETE MIGRATION command, see [“DFSMS DELETE” on page 144](#).
2. If files are listed that are migrated but migrated data does not exist for these files (ML1 or ML2), restore them from the backup of the primary file.
3. If files are listed that are migrated and migrated data is located in both ML1 and ML2, recall and validate the data. If the recall fails or if the file data is incorrect, then that file must be restored from the primary file's backup. For more details on the DFSMS REPORT command, see [“DFSMS REPORT” on page 160](#).
4. If the data in ML1 has been lost, recover any file migrated after the last backup from the primary backup. Any file recalled after the last backup requires no action. For information about recovering a specific file, see [“Recovering Specific Files” on page 85](#).

Recovering From an ML2 Failure

If there is a TSM failure where ML2 data has been lost, consult TSM procedures to restore the TSM data. After restoring ML2 data, you need to resynchronize ML2 with primary storage. In order to do this, perform the steps listed above for ML1 resynchronization.

² Typically, when a volume in an SFS storage group becomes unusable, you cannot recover only that volume. Instead, you need to recover the entire storage group. The procedures documented here for recovery of an entire storage group are the procedures to follow for media failures as well.

Chapter 9. Working with a Minidisk List

The Minidisk Application provides the ability to obtain, organize, and save information about specific minidisks. The saved information is stored in minidisk lists and is easily accessible through either the Minidisk Application or the List Application. The following sections discuss creating, saving, and accessing minidisk lists.

Using ISMF for Minidisk Tasks

After creating a new minidisk list, you can use it immediately, save it for later use, or tailor it by sorting, filtering, or hiding list entries. Tailoring a list allows you to arrange entries in the most convenient order for the task you want to perform. You can also keep a record of the most recent tasks you have performed against the list entries.

Creating a New Minidisk List

To create a new minidisk list, select the minidisk option from the ISMF Primary Option menu. After selecting this option, the Minidisk Selection panel shown in [Figure 47 on page 88](#) is displayed. On this panel you specify the selection criteria that determines which minidisks are displayed. Specifying selection criteria is accomplished by entering a relational operator and a comparison value in the REL OP and VALUE fields. The combination of selection criteria you enter determines the resulting minidisk list (for example, select only those minidisks that reside on a specific volume or only those minidisks that belong to a particular general user). If no selection criteria is specified all minidisks are displayed and the entries are sorted (1) by volume, (2) by starting location on that volume, and (3) by minidisk owner ID.

```

DGTDNSD1                                MINIDISK SELECTION PANEL                Page 1 of 2
COMMAND ===>

SELECT SOURCE OF MINIDISK LIST ===> 2      (1 - saved list, 2 - new list)

1 REUSE A SAVED LIST
  LIST NAME ===>

2 GENERATE A NEW LIST FROM CRITERIA BELOW

                                REL OP   VALUE
                                -----
DEVICE TYPE                     ===>
DEVICE VOLSER                   ===> EQ    FCUSR7
MDISK ADDRESS                   ===>
MDISK OWNERID                   ===>
MDISK SIZE                      ===> GE    3
MDISK START                     ===>

USE ENTER TO PERFORM SELECTION; USE DOWN COMMAND FOR NEXT SELECTION PANEL;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

```

DGTDNSD2                                MINIDISK SELECTION PANEL                Page 2 of 2
COMMAND ===>

TO FURTHER LIMIT THE GENERATED LIST, SPECIFY ONE OR MORE:

                                REL OP   VALUE
                                -----
DUPLEX STATUS                   =====>
DASD FAST WRITE STATUS         =====>
READ CACHE STATUS              =====>
CACHE FAST WRITE STATUS        =====>

USE ENTER TO PERFORM SELECTION; USE UP COMMAND FOR PREVIOUS SELECTION PANEL;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 47. Sample Selection Criteria Entries

Note: Entries on page 2 of the minidisk selection panel are retained. That is, when returning to the minidisk selection panel previous entries are still located on page 2. If these entries are not required you must erase them.

The following are descriptions of the panel fields displayed in the Minidisk Selection panel:

- SELECT SOURCE OF MINIDISK LIST

The valid values are 1 or 2, described as follows:

 - 1, REUSE A SAVED LIST LIST NAME.
 - 2, GENERATE A NEW LIST FROM CRITERIA BELOW.
- LIST NAME. Enter the name of the minidisk saved list.
- REL OP. This is the relational operator field. It can be used in comparison relations to yield a truth value. The relational operators you can use are:

EQ
Equal to

NE
Not equal to

GT
Greater than

GE

Greater than or equal to

LT

Less than

LE

Less than or equal to

Note: **EQ** and **NE** are the only relational operators accepted on page two of the Minidisk Selection panel.

- **VALUE.** This is the primary value against which the comparison is made.
- **DEVICE TYPE.** This is the general name for the device to which the search is limited.
- **DEVICE VOLSER.** This field is the volume serial assigned to the device.
- **MDISK ADDRESS.** This is the address of the virtual minidisk limiting the comparison.
- **MDISK OWNERID.** This is the identification of the minidisk owner.
- **MDISK SIZE.** This is the size of the minidisk in cylinders.
- **MDISK START.** This is the starting cylinder on the real volume where the comparison begins.
- **DUPLEX STATUS.** This is where you specify that disks with a specified duplex status are to be included in your list.
- **DASD FAST WRITE STATUS.** This is where you specify that disks with a specific DASD Fast Write status (for example, active, inactive, or pinned) are to be included in your list.
- **READ CACHE STATUS.** This is where you specify that disks with a specific Read Cache status (for example, active, inactive, or pending) are to be included in your list.
- **CACHE FAST WRITE STATUS.** This is where you specify that disks with a specific Cache Fast Write status (for example, active, inactive, or pinned) are to be included in your list.

Note: An asterisk (*) can be used in character data entry fields on the first page of the Minidisk Selection panel. An asterisk is not valid on page two of the Minidisk Selection panel. If you need to change the selection criteria you have entered, you can type over your original entries, or use the CLEAR command to reset the entry fields to blanks and begin again. Press ENTER after you complete your selection criteria entries.

The Minidisk List panel appears, displaying the first page of entries in your new list. [Figure 48 on page 90](#) shows a sample minidisk list that can result from using the selection criteria indicated in the previous example.

```

DGTLMML1
COMMAND ==>
MINIDISK LIST PANEL
SCROLL ==> HALF
Entries 1-14 of 16
Data Columns 4-7 of 13
ENTER LINE OPERATORS BELOW:

```

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	MDISK START	MDISK END	MDISK SIZE	MDISK GAP
---(1)---	--(2)---	-(3)-	---(4)---	---(5)---	---(6)---	---(7)---
	MAINTDEV	1A1	30	129	100	-----
	MAINTDEV	19F	130	229	100	-----
	MAINTDEV	120	230	479	250	-----
	MAINTDEV	19A	480	879	400	-----
	MAINTDEV	BC1	880	909	30	-----
	MONWRITE	191	1406	1605	200	-----
	CSERV0X0	191	1606	1610	5	-----
	RTMTSTA	191	1665	1694	30	-----
	RTMTSTB	191	1695	1704	10	-----
	RTMTSTC	191	1705	1714	10	-----
	G32TST	191	1715	1744	30	-----
	STUMP	191	1910	1929	20	-----
	BORREG01	191	1930	1949	20	-----
	BEACHMA	191	1950	1958	9	-----

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

DGTLMML1
COMMAND ==>
MINIDISK LIST PANEL
SCROLL ==> HALF
Entries 1-14 of 16
Data Columns 8-12 of 13
ENTER LINE OPERATORS BELOW:

```

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	DEVICE VOLSER	DEVICE TYPE	DUPLEX STATUS	DASD FW STATUS	RD CACHE STATUS
---(1)---	--(2)---	-(3)-	--(8)-	--(9)--	---(10)---	---(11)---	---(12)---
	MAINTDEV	1A1	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	MAINTDEV	19F	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	MAINTDEV	120	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	MAINTDEV	19A	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	MAINTDEV	BC1	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	MONWRITE	191	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	CSERV0X0	191	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	RTMTSTA	191	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	RTMTSTB	191	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	RTMTSTC	191	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	G32TST	191	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	STUMP	191	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	BORREG01	191	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE
	BEACHMA	191	FCUSR7	3390	SIMPLEX	ACTIVE	ACTIVE

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 48. Example of a Minidisk List Panel

Note:

1. This example panel is set to half scroll (default); if needed, you can set the scroll to PAGE.
2. When the Minidisk List panel is displayed only columns 1–7 appear, you must scroll right in order to view the remaining columns.

The following is a description of the panel fields displayed in the Minidisk List panel:

- LINE OPERATOR (1). This field is where the ISMF line operators (see [“Using ISMF Commands and Line Operators”](#) on page 162) are entered to begin processing. This is the only input field on this panel.
- MDISK OWNERID (2). This is the identification of the minidisk owner.
- MDISK ADDR (3). This field is the address of the minidisk.
- MDISK START (4). This field is the starting cylinder on the real volume.
- MDISK END (5). This field identifies the ending cylinder on the real volume.
- MDISK SIZE (6). This field indicates the size of the minidisk in cylinders.
- MDISK GAP (7). This field indicates the number of cylinders between contiguous minidisks. The following special characters may appear in this data column:

If the value is not available

??????????

If the value cannot be displayed due to an error or for a full-pack defined minidisk the gap field contains the question marks.

- DEVICE VOLSER (8). This field is the volume serial assigned to the device.
- DEVICE TYPE (9). This field is the general name for the device to which the search is to be limited.
- DUPLEX STATUS (10). This field indicates the duplex status of the volume where this minidisk resides.
- DASD FW STATUS (11). This field indicates the status of DASD Fast Write on the volume where this minidisk resides.
- RD CACHE STATUS (12). This field indicates the status of Read Cache on the volume where this minidisk resides.
- CACHE FW STATUS (13). This field indicates the status of Cache Fast Write on the volume where this minidisk resides (scroll right to view this field).

One page of list entries is displayed at a time. You can enter a scroll amount in the SCROLL field at the top of the panel, such as PAGE, HALF, or a specific number of entries, then use the UP, DOWN, LEFT, or RIGHT commands to view different sections of the minidisk list.

If you issue moves or checks from a list or any line operators that alter the list the Minidisk List Save panel is displayed (shown in [Figure 49 on page 91](#)).

```
DGTDSV01          MINIDISK LIST SAVE PANEL
COMMAND ==>

YOU HAVE MADE CHANGES TO THE DISPLAYED LIST SINCE ENTERING THE
MINIDISK APPLICATION.  YOU CAN SAVE THE LIST BEFORE YOU EXIT.


DO YOU WANT TO SAVE ==> N (Enter Y or N)


LIST NAME          ==>


REPLACE LIST IF IT
  ALREADY EXISTS   ==> N (Enter Y or N)


USE ENTER TO CONTINUE; USE CANCEL COMMAND TO RETURN TO MINIDISK LIST;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 49. Example of a Minidisk List Save Panel

Saving a Minidisk List

You can save a minidisk list in its entirety, or save a subset of the list. To save a minidisk list, create the list you want to save and then type SAVE and the 1–8 character name you want to assign to your list on the command line of the ISMF Minidisk List panel.

- ISMF does not allow you to use DGT, ISP, or ISR as the first three characters of a list name. DGT, ISP, and ISR are reserved characters.
- When you save a list, all the entries in the list are saved in their displayed order. Entries scrolled off the screen are saved, but entries excluded by the FILTER command and the HIDE line operator are *not* saved.

Line operator history, including messages and request IDs, is also saved with the list so that you can review the tasks you have performed and obtain information to use the QUERY and DISCARD commands for particular requests.

You can save a list or a subset of the list to:

- Submit move or check requests at a more convenient time.

- Use it to request status information about move or check requests you have issued.
- Use the entries in the list as input to a REXX EXEC that produces a formatted report.
- Keep a record of the most recent tasks performed against specific minidisks.

Figure 50 on page 92 shows a sample entry to save a list called LISTA.

```
DGTLNML1      MINIDISK LIST PANEL
COMMAND ==> SAVE LISTA
ENTER LINE OPERATORS BELOW:

                                SCROLL ==> HALF
                                Entries 1-14 of 15
                                Data Columns 4-9 of 13
```

Figure 50. Sample Entry to Save a List

If a list by the name you specified already exists and you want to replace it with a more current version, type SAVE with the REPLACE option as shown in Figure 51 on page 92.

```
DGTLNML1      MINIDISK LIST PANEL
COMMAND ==> SAVE LISTA REPLACE
ENTER LINE OPERATORS BELOW:

                                SCROLL ==> HALF
                                Entries 1-14 of 15
                                Data Columns 4-9 of 13
```

Figure 51. Sample Entry to Replace a Saved List

Accessing a Saved Minidisk List

After saving a minidisk list, you access it through either the Minidisk Application or the List Application.

To access a saved list through the Minidisk Application:

1. Select the Minidisk Application from the ISMF Primary Option menu. The Minidisk Selection panel appears. The panel is shown in Figure 47 on page 88 along with panel field descriptions.
2. Type **1** in the Select Source of Minidisk List field, then type the name of the minidisk list you want to use in the List Name field.

You can also access a saved list through the List Application by entering **LIST** in the line operator field next to the name of the minidisk list you want to use. The Minidisk List panel appears, displaying the list you selected. When you access a saved minidisk list through the List Application, you automatically enter the Minidisk Application. You can then issue the same commands and line operators as if you had entered the Minidisk Application directly. For additional information about the List Application, see “List Application” on page 30.

You might want to share saved lists among ISMF users in your installation. You can use both your own saved minidisk lists and minidisk lists created and saved by other ISMF users.

If your ISMF session uses a MACLIB where lists belonging to other ISMF users are stored, you can access those lists automatically during your ISMF session. You can also have the lists sent to your user ID and read them into MACLIBs you use. If your ISMF session uses several MACLIBs, they might contain minidisk lists that have the same list name. For minidisk lists with duplicate names, only the first occurrence encountered is shown in the list of saved ISMF lists.

You can write EXECs to access and extract information from saved ISMF lists. For more information about accessing a saved list through an EXEC, see “Accessing a Saved List through EXECs” on page 173.

Chapter 10. Moving and Checking CMS Minidisks

DFSMS/VM includes many functions and features that make moving CMS minidisks to new or other DASD faster and easier. For this process you need to plan the move, implement the move, and verify that the operation is successful.

The following sections are divided into three primary topic areas:

- Planning move activities
- Implementing the move process
- Verifying the integrity of CMS minidisks

These sections provide information to help you determine whether any further customization is required, and to help you plan your move activities. These sections also provide information about the various commands and line operators that you will use during the move process, including ways to verify the integrity of the minidisks.

Planning Move Activities

Planning is the most important phase of the move process. This section contains information to help you plan your move activities. Topics include:

- Customization considerations
- Grouping minidisks for move requests
- Selecting move options
- Issuing move requests
- Tracking move requests

Customization Considerations

First consider whether DFSMS/VM has been customized appropriately for you to move minidisks. You also must consider whether you need to modify the installation-wide exits provided with the DFSMS/VM feature and any of the defaults specified in the DFSMS/VM control file (DGTVCNTL DATA in the SFS directory VMSYS:DFSMS.CONTROL).

Refer to the [*z/VM: DFSMS/VM Customization*](#) for descriptions of the DFSMS/VM control file variables.

Grouping Minidisks for Move Requests

To ensure a timely and orderly move to new DASD, consider the best way to group your minidisks for move requests and what type of minidisks you are moving. If you are installing numerous new DASDs, the move process can be expedited by defining several minidisk servers.

Note: If DIRMAINT usage is high, increase the directory timeout value in the DFSMS/VM control file. If this change is not made, a DIRMAINT request can fail due to time surpassing the timeout value.

A logical first grouping is by volume for CMS minidisks. However, you might also want to start with groups sorted by minidisk owner ID or minidisk size, depending on what is most logical and efficient for your operation. You can create appropriate minidisk lists by using the Minidisk Application and by choosing from any combination of ten different selection criteria.

After you create your initial minidisk lists, you can separate them into smaller, more manageable groups of minidisks. You can divide your minidisks into groups that represent the number of minidisks you want to track at one time and save these lists. This can be done by using the FILTER and SORT commands.

Note: Separating them into smaller groups of minidisks is not necessary. For example, if you are moving minidisks to a new DASD, it is preferable to move all of the minidisks at one time.

You can sort the initial list by using the SORT command to help you decide what the logical groupings are for each volume. You can also eliminate any minidisks by using the HIDE line operator before or after you group the minidisks.

No matter how you decide to group your minidisks, you may create and save each list until you have organized all the minidisks you plan to move. When you are ready to move the minidisks identified in each list, you may access the lists through the List Application. The List Application presents a list of all your saved minidisk lists. By saving a minidisk list you can save yourself some time when you reuse the list.

You need to select another method to move minidisks that are in any of these categories:

- Non-CMS minidisks that belong to the control program (CP)
- Minidisks that belong to service machines
- Guest minidisks (for example, OS or VSE minidisks)
- S and Y minidisks
- Minidisks that are on a 3390-A Extended Address Volume (EAV), or are being moved to an EAV that is larger than 65520 cylinders.

To help you select a way to move these types of minidisks, see the CMS documentation for your installation.

Selecting Move Options

This section provides information to help you decide which options are appropriate for moving CMS minidisks. The commands for moving CMS minidisks are the ISMF MOVE line operator, ISMF MOVE command, DFSMS MOVE or the DFSMS COPY command.

When you issue move requests, ISMF MOVE and DFSMS MOVE offer you several move options:

- Determine whether you want to specify target volumes or target groups.
- If you specify target volumes, determine whether you want to specify target device type and target starting cylinder. (If you specify one, you must also specify the other.)
- Determine the appropriate method to use to specify the target minidisk size. You can specify:
 - A different target minidisk size in cylinders for each minidisk move or each minidisk list
 - A different percentage of the source minidisk size for each minidisk move or minidisk list

Consider using the default value of 100% so that a move can be submitted quickly on an entire minidisk list. Using 100% ensures that the target minidisk contains at least as much space as the source minidisk, at the source minidisk block size.

- Determine whether the target minidisk label should contain the original creation date or the date-of-move creation date.
- Determine the appropriate move pacing level to use based on when you plan to issue your move requests and what effect you want your move activities to have on overall system performance.

The move pacing level option applies only when DFSMS COPY is used to perform the format and copy steps of the move process.

- Determine the appropriate I/O buffer size based on when you plan to issue your move requests and what effect you want your move activities to have on overall system performance.

The I/O buffer size option applies only when DFSMS COPY is used to perform the format and copy steps of the move process.

- Consider continuing a move operation if a permanent error occurs.
- Consider reblocking your files *after* you move your minidisks. If you reblock, DFSMS COPY cannot be used to perform the format and copy steps of the move process. CMS FORMAT and CMS COPYFILE will be used.

Note: A reserved minidisk *cannot* be reblocked. If an attempt is made to reblock a reserved minidisk, DFSMS/VM returns an error message.

If you specify a percentage for target minidisk size and also use the reblock files option, there is no guarantee that the resulting target minidisk will be of sufficient size. Your move requests can fail.

DFSMS COPY is the high-performance data mover included with DFSMS/VM. When you issue a move request through ISMF or DFSMS MOVE, DFSMS COPY performs the format and copy steps of the move process unless a restriction applies (packing, unpacking, or reblocking causes this restriction). If a restriction applies, CMS FORMAT and CMS COPYFILE perform those steps.

You can invoke DFSMS COPY independently under CMS to copy CMS file system minidisks or other types of CMS-formatted minidisks. For example, you can use DFSMS COPY to copy minidisks that contain Shared File System (SFS) or Structured Query Language/Data System (SQL/DS) data.

When you run DFSMS COPY under CMS, it performs only the format and copy steps involved in moving minidisks. It does not perform any of the automated steps provided through ISMF or DFSMS MOVE. For example, if you use DFSMS COPY to move CMS-formatted minidisks, you must manually do the directory updates.

See [“DFSMS COPY” on page 141](#) for command syntax and additional information.

You can invoke DFSMS MOVE independently under CMS to MOVE CMS file system minidisks. DFSMS MOVE must be issued for each individual minidisk. It does not accept a list of minidisks. You may use DFSMS MOVE in an EXEC to achieve the equivalent of list processing. See [“DFSMS MOVE” on page 151](#) for command syntax and additional information.

ISMF includes the MOVE list command and the MOVE line operator. You use them to issue requests against CMS minidisks identified in a minidisk list. DFSMS/VM assigns a request ID to each request that is successfully accepted and queues the request for processing. See [“Implementing the MOVE Process” on page 96](#) for detailed information about the move process.

Issuing Move Requests

After you have grouped your CMS minidisks into logical and manageable groups, and have selected which options to use, you are ready to issue your move requests.

To move the minidisks smoothly and efficiently, with as little disruption as possible to other operations and VM users, you need to determine the best time to issue those requests.

The following are considerations to keep in mind:

- If you issue move requests during prime shift:
 - Be sure to select move options that help reduce the impact on your other operations, such as a smaller I/O buffer size option or an intermittent move pacing level option.
 - Be aware that move requests may fail because server virtual machines are not able to obtain exclusive links to minidisks due to increased general user access.

Note: If DIRMAINT usage is high, you should increase the directory timeout value in the DFSMS/VM control file. If this change is not made, a DIRMAINT request can fail due to time surpassing the timeout value.

- If you issue move requests during off hours:
 - Consider increasing the number of servers to improve throughput.
 - Consider increasing the I/O buffer size and using a continuous move pacing level to maximize DFSMS/VM performance.
 - Consider modifying the DGTQLINK installation-wide exit to force general users to relinquish access to minidisks. A move request fails if the server virtual machine cannot obtain an exclusive link to the minidisk.

Tracking Move Requests

After you have issued move requests, you may track their progress and determine whether they have completed successfully. There are several ways to obtain status information about requests.

You can use the ISMF QUERY command or the ISMF QUERY line operator to check on the status of move requests. The DFSMS QUERY command may also be used. You may save the minidisk list after you issue move requests from that list. Line operator history, including the request ID for each move request, is saved with the list. That allows you to use the list later to obtain request IDs and request status information.

You can perform a quick check to determine if all the minidisks in a list were moved by creating a new minidisk list using the expected results as selection criteria.

DFSMS QUERY or ISMF QUERY can be used to obtain the status of an entire MOVE list. The response you receive will be determined by the options you used for the DFSMS QUERY command.

You will receive a DFSMS response file when the MOVE request completes. This file will contain messages which indicate whether the MOVE was successful or failed. If the MOVE was successful, the location of the new minidisk will be provided.

For information about obtaining a request ID, see [“Request IDs” on page 110](#). For information about using the QUERY command, see [“Using ISMF Commands and Line Operators” on page 162](#). See [“DFSMS QUERY” on page 157](#) for command syntax and additional information.

Implementing the MOVE Process

You use a minidisk list to issue requests to move multiple CMS minidisks from one location to another. For example, you might want to move a minidisk from one volume to another, or move all of your CMS minidisks to new DASD. The following sections provide step-by-step instructions to issue MOVE requests.

To move CMS minidisks:

- Use the ISMF MOVE line operator to issue move requests for individual minidisks. You can:
 - Issue a move request for a single minidisk
 - Repeat the line operator to issue move requests for multiple minidisks and choose different move options for each request
 - Use the line operator in last-use mode to issue move requests for multiple minidisks, using the same move options for each request
- Use the ISMF MOVE list command to issue move requests for all the minidisks identified in a minidisk list, using the same move options for each request.
- Use the command line DFSMS MOVE to implement the move process without ISMF.

Move Options

When you issue a move request using either the MOVE line operator or the MOVE list command, ISMF displays entry panels that allow you to choose move options. If you use the MOVE line operator in last-use mode for multiple minidisks or use the MOVE list command, your options apply to each move request issued.

Source Minidisk Options

There are options that allow you to control whether a minidisk can be moved if it contains IPL text, if there is a CP/CMS size mismatch, if the minidisk is reserved, or if it is based on the minidisk link mode. However, those options are dependent upon limits specified in the DFSMS/VM control file.

Each of those options has a corresponding keyword-value pair in the DFSMS/VM control file. The keywords are:

```
MINIDISK_IPLTEXT_LIMIT
MINIDISK_MISMATCH_LIMIT
MINIDISK_RESERVED_LIMIT
MINIDISK_LINK_MODE_LIMIT
```

If the value specified as the limit for the first three keywords is N, you cannot override that value by specifying Y for the corresponding option on the Move Entry panel. However, if the value specified as the limit is Y, you can specify N on the Move Entry panel to override it.

Table 5 on page 97 lists the values for the fourth keyword MINIDISK_LINK_MODE_LIMIT and shows the relationship and results of the panel values in conjunction with the DFSMS/VM control file values.

Table 5. MINIDISK_LINK_MODE_LIMIT Value Relationships

Panel Value	Control File Value	DFSMS/VM Action
1	1	No links to source minidisk permitted
1	2	No links to source minidisk permitted
1	3	No links to source minidisk permitted
2	1	Move fails, control file override not allowed
2	2	Read links to source minidisk permitted
2	3	Read links to source minidisk permitted
3	1	Move fails, control file override not allowed
3	2	Move fails, control file override not allowed
3	3	Permit any links to source minidisk

It is recommended that this keyword value be set to 1 to maintain minidisk integrity.

Note: The options shown above can be specified in either the DFSMS/VM control file or in the ISMF panel. The only time an ISMF panel option can override one in the DFSMS/VM control file is when the control file allows it.

Within the ISMF panels the remaining options related to the source minidisk are:

- MOVE IF EMPTY MINIDISK

Choosing whether to move a minidisk if it contains no files

- NUMBER OF RETRIES FOR LINK

Indicating the number of times a server virtual machine attempts to relink to a minidisk

- LINK RETRY INTERVAL

Indicating the time interval between each attempt to relink to a minidisk

- FORMAT SOURCE AFTER MOVE

Specifying whether the source minidisk is cleared after it has been successfully moved to the target minidisk

Target Minidisk Options

When the MOVE line operator or the MOVE list command is issued, you have the option of specifying:

Target volume or target group

Target duplex status

Target minidisk size

Additionally the target device type and starting location can be specified with the MOVE line operator.

You must specify either a target volume or a target group. If you specify a target volume with the MOVE line operator, you can specify the target device type and starting location options (you must specify both together).

If you specify a target volume with no options, the target minidisk is allocated on the volume you specify, and the starting location is chosen by your directory maintenance product (for example, DIRMAINT).

If you specify a target group, the target minidisk is allocated on a volume included in the group of volumes you specify. The specific volume and starting location is chosen by your directory maintenance product.

Note: If you specify target group, you must have previously defined the target group to your directory maintenance product.

If you specify target duplex status, the allocated target minidisk is checked for the requested status. You can enter a value of simplex, duplex, or leave the selection blank. If you leave the selection blank, the check of the requested value is bypassed. If the allocated target minidisk does not match the requested value, the MOVE fails.

The target minidisk size is required for both the MOVE line operator and the MOVE list command. You can specify a target minidisk size as a specific number of cylinders, or as a percentage of the source minidisk size. Or, you can accept the default size of 100% that is supplied by ISMF.

Other Move Options

You can use additional options to:

- Change the minidisk creation date
- Continue moving a minidisk when feasible for certain types of errors (applies only when DFSMS COPY is used)
- Change the block size of files on a minidisk (forces use of CMS COPYFILE)
- Pack or unpack all files on a minidisk (forces the use of CMS COPYFILE)

You can also select the following:

- A pacing level for the move operation (continuous or intermittent)
- An I/O buffer size (for CKD devices — 1, 2, or 5 tracks or 1 cylinder).

Move Process Results

The automated move process for the ISMF MOVE line operator, the ISMF MOVE list command, and the command line DFSMS MOVE performs the following tasks:

- Locates the target minidisks
- Ensures that the cylinders for the target minidisks are formatted
- Physically copies the minidisks
- Releases (and optionally clears) space occupied by the source minidisks
- Performs object directory updates in place (for CP and your directory maintenance product)
- Adjusts the sizes of the allocation maps for the target minidisks to reflect the new sizes in blocks

The move process uses DFSMS COPY whenever possible to perform the format and copy steps of the move process. If you specify DFSMS MOVE with a PACK, UNPACK or REBLK option then it will prohibit the use of the high speed data mover and use CMS COPYFILE and CMS FORMAT instead. DFSMS COPY does not support reblocking. If you choose the option to change the block size of files on a minidisk, DFSMS or the minidisk server will then use CMS FORMAT and CMS COPYFILE.

For more information about DFSMS COPY functions and restrictions, see “DFSMS COPY” on page 141. For more information about CMS FORMAT and CMS COPYFILE, see the CMS command reference book appropriate for your installation.

Move Errors

If an error occurs while a move request is being processed, you do not receive error messages through ISMF. Messages related to the move process are returned to you in a DFSMS response file.

For information about possible causes of move failures and instructions to obtain and correct errors, see [*z/VM: DFSMS/VM Diagnosis Guide*](#)

Using the MOVE Line Operator

To use the MOVE line operator to issue move requests:

1. Create or display the minidisk list that identifies the CMS minidisks you want to move.
2. On the Minidisk List panel, type **MOVE** in the line operator field next to the first (or only) CMS minidisk you want to move. See [Figure 52 on page 99](#).

```
DGTLNML1          MINIDISK LIST PANEL
COMMAND ==>>>          SCROLL ==>> HALF
                        Entries 1-14 of 16
                        Data Columns 4-7 of 13

ENTER LINE OPERATORS BELOW:

  LINE   MDISK   MDISK   MDISK   MDISK   MDISK   MDISK
  OPERATOR OWNERID ADDR   START   END     SIZE   GAP
  --- (1) --- -- (2) --- -- (3) --- -- (4) --- -- (5) --- -- (6) --- -- (7) ---
  MOVE    MAINTDEV 1A1      30      129     100      0
          MAINTDEV 19F      130     229     100      0
          MAINTDEV 120      230     479     250      0
          MAINTDEV 19A      480     879     400      0
          MAINTDEV BC1      880     909      30     496
          MONWRITE 191     1406    1605     200      0
          CSERV0X0 191     1606    1610      5      54
          RTMTSTA  191     1665    1694      30      0
          RTMTSTB  191     1695    1704      10      0
          RTMTSTC  191     1705    1714      10      0
          G32TST   191     1715    1744      30     165
          STUMP    191     1910    1929      20      0
          BORREG01 191     1930    1949      20      0
          BEACHMA  191     1950    1958       9      0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 52. Entering the MOVE Line Operator

If you are moving a single minidisk, go to step 3. If you are moving multiple minidisks, continue with the procedure that follows:

Repeating the MOVE Line Operator:

Repeat the MOVE line operator when you want to move more than one minidisk and when you want to choose different move options for each request (for example, when you want to specify a different target minidisk size for each minidisk you move).

When you repeat a MOVE line operator, the Move Entry panels are displayed once for each request.

To repeat the MOVE line operator:

- a. Type an equal sign (=) next to any subsequent entries in the list that identify minidisks you want to move as shown in [Figure 53 on page 99](#).

```
DGTLNML1          MINIDISK LIST PANEL
COMMAND ==>>>          SCROLL ==>> HALF
                        Entries 1-14 of 16
                        Data Columns 4-7 of 13

ENTER LINE OPERATORS BELOW:

  LINE   MDISK   MDISK   MDISK   MDISK   MDISK   MDISK
  OPERATOR OWNERID ADDR   START   END     SIZE   GAP
  --- (1) --- -- (2) --- -- (3) --- -- (4) --- -- (5) --- -- (6) --- -- (7) ---
  MOVE    MAINTDEV 1A1      30      129     100      0
  =       MAINTDEV 19F      130     229     100      0
  =       MAINTDEV 120      230     479     250      0
  =       MAINTDEV 19A      480     879     400      0
  =       MAINTDEV BC1      880     909      30     496
  =       MONWRITE 191     1406    1605     200      0
  =       CSERV0X0 191     1606    1610      5      54
  =       RTMTSTA  191     1665    1694      30      0
  =       RTMTSTB  191     1695    1704      10      0
  =       RTMTSTC  191     1705    1714      10      0
  =       G32TST   191     1715    1744      30     165
  =       STUMP    191     1910    1929      20      0
  =       BORREG01 191     1930    1949      20      0
  =       BEACHMA  191     1950    1958       9      0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 53. Repeating the MOVE Line Operator

b. Go to step 3.

Using the MOVE Line Operator in Last-Use Mode:

Use the MOVE line operator in last-use mode when you want to move more than one minidisk and when you want to use the same move options for each request. When you use last-use mode, the Move Entry panels are displayed only once.

If you want to issue a move request for the entire list of minidisks, and you want to use the same move options for each request, use the MOVE list command, described in [“Using the MOVE List Command” on page 102](#) or use the line operator as demonstrated in [Figure 53 on page 99](#).

To use the MOVE line operator in last-use mode:

- Type == (two equal signs) in the line operator field next to the second entry that identifies a minidisk you want to move.
- Type = (a single equal sign) in the line operator field for each subsequent entry that identifies a minidisk you want to move.

[Figure 54 on page 100](#) shows sample entries to use the MOVE line operator in last-use mode.

DGTLNML1
COMMAND ===>

MINIDISK LIST PANEL

SCROLL ===> HALF
Entries 1-14 of 16
Data Columns 4-7 of 13

ENTER LINE OPERATORS BELOW:

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	MDISK START	MDISK END	MDISK SIZE	MDISK GAP
---(1)---	---(2)---	---(3)---	---(4)---	---(5)---	---(6)---	---(7)---
MOVE	MAINTDEV	1A1	30	129	100	0
==	MAINTDEV	19F	130	229	100	0
=	MAINTDEV	120	230	479	250	0
=	MAINTDEV	19A	480	879	400	0
=	MAINTDEV	BC1	880	909	30	496
=	MONWRITE	191	1406	1605	200	0
=	CSERV0X0	191	1606	1610	5	54
=	RTMTSTA	191	1665	1694	30	0
=	RTMTSTB	191	1695	1704	10	0
=	RTMTSTC	191	1705	1714	10	0
=	G32TST	191	1715	1744	30	165
	STUMP	191	1910	1929	20	0
	BORREG01	191	1930	1949	20	0
	BEACHMA	191	1950	1958	9	0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 54. Using the MOVE Line Operator in Last-Use Mode

c. Go to step 3.

3. Press **ENTER**.

Page 1 of the Move Entry panel appears. [Figure 55 on page 101](#) shows the defaults that appear on that panel.

If you repeated the line operator or used last-use mode, the source minidisk owner ID, minidisk address, and minidisk size shown on the Move Entry panel are those associated with the first request you entered.

4. Fill in page 1 of the Move Entry panel or leave the default values as shown.

While the Move Entry panel is displayed, press **PF1** to select help panels that contain detailed information about each field.

5. Press **ENTER**.

Page 2 of the Move Entry panel appears. [Figure 55 on page 101](#) shows the defaults that appear on that panel.

```

DGTDM01                                MOVE ENTRY PANEL                                Page 1 of 2
COMMAND ===>

SOURCE MINIDISK:  MAINTDEV  1A1  SOURCE MINIDISK SIZE:  100

OPTIONALLY SPECIFY ONE OR MORE:

TARGET MINIDISK TYPE ===> 1                (1 for CMS)

MOVE IF IPL TEXT PRESENT    ===> N        (Y or N)
MOVE IF CP/CMS SIZE MISMATCH ===> N        (Y or N)
MOVE IF RESERVED MINIDISK   ===> N        (Y or N)
MOVE IF EMPTY MINIDISK      ===> Y        (Y or N)

ALLOWABLE EXISTING LINKS    ===> 1        (1=NONE, 2=READ, 3=WRITE)
NUMBER OF RETRIES FOR LINK   ===> 2        (0 to 99)
LINK RETRY INTERVAL  HOURS  ===> 00      MIN ===> 00      SEC ===> 10

FORMAT SOURCE AFTER MOVE =====> N      (Y or N)

USE ENTER TO CONTINUE.
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

```

DGTDM02                                MOVE ENTRY PANEL                                Page 2 of 2
COMMAND ===>

SOURCE MINIDISK:  MAINTDEV  1A1  SOURCE MINIDISK SIZE:  100

SPECIFY ONE OF THE FOLLOWING:
TARGET GROUP                ===>          (Group name)
TARGET VOLUME                ===>          (Volser name)
FOR TARGET VOLUME, OPTIONALLY SPECIFY BOTH:
  TARGET DEVICE TYPE          ===>          (For example, 3380)
  TARGET LOCATION             ===>          (Starting location)
OPTIONALLY SPECIFY THE FOLLOWING:
TARGET VOLUME DUPLEX STATUS  ===>          (blank, SIMPLEX, DUPLEX)
TARGET MINIDISK SIZE         ===> 100%     (Num. of cyls or % of source)
MINIDISK CREATION DATE       ===> 2        (1=Today's date, 2=Old date)
MOVE PACING LEVEL            ===> 2        (1=Continuous, 2=Intermittent)
I/O BUFFER SIZE              ===> 3        (1=1trk,2=2trk,3=5trk,4=1cyl)
CONTINUE ON PERMANENT ERROR  ===> N        (Y or N)
PACK OPTION                  ===> 3        (1=Pack, 2=Unpack, 3=No pack)
REBLOCK FILES                ===> N        (Y or N)
REBLOCK SIZE                 ===>          (512,1024,2048,4096)

USE ENTER TO CONTINUE.
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 55. Move Entry Panel

6. Fill in page 2 of the Move Entry panel with the target and additional move options, or leave the defaults as shown.

You must specify either a TARGET VOLUME or a TARGET GROUP.

- If you specify a TARGET VOLUME and TARGET DEVICE TYPE then a TARGET LOCATION is also required.
- TARGET DEVICE TYPE and TARGET LOCATION can only be specified if you specify TARGET VOLUME. And if either TARGET DEVICE TYPE or TARGET LOCATION is used then they both must be used.
- If you specify TARGET DUPLEX STATUS, you must choose one of the following values:
 - Blank—target minidisk can be on a simplex or duplex volume
 - Simplex—target minidisk must be on a simplex volume
 - Duplex—target minidisk must be on duplex volume
- For TARGET MINIDISK SIZE, you can specify either a number of cylinders, or a percentage of the source minidisk capacity. When you specify a percentage, the number of cylinders allocated for the target minidisk is sufficient to hold the specified percentage of source minidisk cylinders.

When you want to specify a percentage, be sure to type the percent sign (%). Otherwise, the value you enter will be interpreted as a number of cylinders.

- For PACK OPTION, if you do not change the default of 3, DFSMS COPY is used when no restrictions apply. Choose option 1 if you want to use CMS COPYFILE with a PACK parameter, or option 2 if you want to use CMS COPYFILE with an UNPACK parameter.

Note: If you specify a percentage for TARGET MINIDISK SIZE and also select the REBLOCK FILES option, there is no guarantee that the resulting target minidisk will be of sufficient size. Such a move request could fail.

While the Move Entry panel is displayed, press **PF1** to select help panels that contain detailed information about each field on this panel.

7. Press **ENTER**.

- If you issue one move request, the Minidisk List panel is displayed. The line operator field contains line operator history. The top right corner of the panel contains a short message, such as MOVE ISSUED OK.
- If you repeat the MOVE line operator, one Move Entry panel appears for each request you issue. Fill in the panels for each request.

When you press ENTER after filling in the panels for the last request, the Minidisk List panel is displayed. The line operator fields contain line operator history for each request. The top right corner of the panel contains a short message related to the last request processed.

- If you use the MOVE line operator in last-use mode, the line operator fields for each request contain line operator history. The top right corner of the panel contains a short message related to the last request processed.

Using the MOVE List Command

You can use the MOVE list command to move all the minidisks identified in a particular minidisk list. However, minidisks that have been excluded from the minidisk list by using the FILTER command or HIDE line operator are not issued.

When you use the MOVE list command, ISMF checks the minidisk owner ID, minidisk address, and minidisk size fields to determine whether they contain valid information. If an entry contains invalid information, a move request cannot be issued for that entry.

If a minidisk list contains any entries with invalid information, ISMF displays a confirmation panel that tells you how many entries contain invalid information, and how many move requests can be issued. You can cancel all requests or issue requests for all valid entries.

To use the MOVE list command to issue move requests:

1. Create or display the minidisk list that identifies the CMS minidisks you want to move.
2. Type **MOVE** on the command line of the Minidisk List panel, as shown in [Figure 56 on page 103](#).

```

DGTLM1
COMMAND ==> MOVE
MINIDISK LIST PANEL
SCROLL ==> HALF
Entries 1-14 of 16
Data Columns 4-7 of 13
ENTER LINE OPERATORS BELOW:

```

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	MDISK START	MDISK END	MDISK SIZE	MDISK GAP
---(1)---	--(2)---	-(3)-	---(4)---	---(5)---	---(6)---	---(7)---
	MAINTDEV	1A1	30	129	100	0
	MAINTDEV	19F	130	229	100	0
	MAINTDEV	120	230	479	250	0
	MAINTDEV	19A	480	879	400	0
	MAINTDEV	BC1	880	909	30	496
	MONWRITE	191	1406	1605	200	0
	CSERVOX0	191	1606	1610	5	54
	RTMTSTA	191	1665	1694	30	0
	RTMTSTB	191	1695	1704	10	0
	RTMTSTC	191	1705	1714	10	0
	G32TST	191	1715	1744	30	165
	STUMP	191	1910	1929	20	0
	BORREG01	191	1930	1949	20	0
	BEACHMA	191	1950	1958	9	0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 56. Entering the MOVE List Command

3. Press **ENTER**.

Page 1 of the Move Entry panel appears. [Figure 57 on page 104](#) shows the defaults that appear on that panel.

4. Fill in page 1 of the Move Entry panel with the move options you want to use, or leave the default values as shown.

While the Move Entry panel is displayed, press **PF1** to select help panels that contain detailed information about each field.

5. Press **ENTER**.

Page 2 of the Move Entry panel appears. [Figure 57 on page 104](#) shows the defaults that appear on that panel.

DGTDNMM1 MOVE COMMAND ENTRY PANEL Page 1 of 2
 COMMAND ==>

OPTIONALLY SPECIFY ONE OR MORE FOR 16 MINIDISKS:

TARGET MINIDISK TYPE ==> 1 (1 for CMS)

MOVE IF IPL TEXT PRESENT ==> N (Y or N)
 MOVE IF CP/CMS SIZE MISMATCH ==> N (Y or N)
 MOVE IF RESERVED MINIDISK ==> N (Y or N)
 MOVE IF EMPTY MINIDISK ==> Y (Y or N)

ALLOWABLE EXISTING LINKS ==> 1 (1=NONE, 2=READ, 3=WRITE)
 NUMBER OF RETRIES FOR LINK ==> 2 (0 to 99)
 LINK RETRY INTERVAL HOURS ==> 00 MIN ==> 00 SEC ==> 10

FORMAT SOURCE AFTER MOVE =====> N (Y or N)

USE ENTER TO CONTINUE.
 USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

DGTDNMM2 MOVE COMMAND ENTRY PANEL Page 2 of 2
 COMMAND ==>

SPECIFY ONE OF THE FOLLOWING CMS TARGET OPTIONS FOR 16 MINIDISKS:

TARGET GROUP ==> (Group name)
 TARGET VOLUME ==> (Volser name)

OPTIONALLY SPECIFY THE FOLLOWING:

TARGET VOLUME DUPLEX STATUS ==> (blank, SIMPLEX, DUPLEX)
 TARGET MINIDISK SIZE ==> 100% (Num. of cyls or % of source)
 MINIDISK CREATION DATE ==> 2 (1=Today's date, 2=Old date)
 MOVE PACING LEVEL ==> 2 (1=Continuous, 2=Intermittent)
 I/O BUFFER SIZE ==> 3 (1=1trk, 2=2trk, 3=5trk, 4=1cyl)
 CONTINUE ON PERMANENT ERROR ==> N (Y or N)
 PACK OPTION ==> 3 (1=Pack, 2=Unpack, 3=No pack)
 REBLOCK FILES ==> N (Y or N)
 REBLOCK SIZE ==> (512, 1024, 2048, 4096)

USE ENTER TO CONTINUE.
 USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 57. Move Command Entry Panel

6. Fill in page 2 of the Move Entry panel.

You must specify either a TARGET GROUP or a TARGET VOLUME.

- If you specify TARGET DUPLEX STATUS, you must choose one of the following values:
 - Blank—target minidisk can be on a simplex or duplex volume.
 - Simplex—target minidisk must be on a simplex volume.
 - Duplex—target minidisk must be on duplex volume.
 - For TARGET MINIDISK SIZE, you can specify either a number of cylinders or a percentage of the source minidisk capacity. When you specify a percentage, the number of cylinders allocated for the target minidisk is sufficient to hold the specified percentage of source minidisk cylinders.
- When you want to specify a percentage, be sure to type the percent sign (%). Otherwise, the value you enter will be interpreted as a number of cylinders.
- For PACK OPTION, leave the default of 3 if you want to use the DFSMS COPY data mover if no restrictions apply. Choose option 1 if you want to use CMS COPYFILE with a PACK parameter, or option 2 if you want to use CMS COPYFILE with an UNPACK parameter.

Note: If you specify a percentage for TARGET MINIDISK SIZE and also select the REBLOCK FILES option, there is no guarantee that the resulting target minidisk will be of sufficient size. Such a move request could fail.

While the Move Entry panel is displayed, press **PF1** to select help panels that contain detailed information about each field.

7. Press **ENTER**.

If all entries in the list contain valid information, ISMF issues a move request for each minidisk identified in the list. The Minidisk List panel reappears. Each line operator field contains line operator history, such as, *MOVE. The top, right corner of the panel contains a short message, such as REQUEST ID:3474.

If any entries in the list contain invalid information, ISMF displays the Move Confirmation panel, shown in [Figure 58 on page 105](#).

```

                                MOVE CONFIRMATION PANEL
COMMAND ==>>

      5  MINIDISK MOVE REQUESTS OF 50  CONTAIN ERRORS.
THE REMAINING 45  MINIDISK MOVE REQUESTS CAN BE ISSUED.

      SHOULD THEY BE ISSUED ? ==> N      (Y or N)

USE ENTER TO CONTINUE;
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.
```

Figure 58. Move Confirmation Panel

8. Enter **Y** to issue requests for all entries that contain valid information.

Enter **N** to cancel all move requests for the entire list.

9. Press **ENTER**.

- If you enter Y on the Move Confirmation panel, ISMF issues move requests for the number of entries indicated on that panel. The Minidisk List panel is displayed.

Each line operator field associated with an entry for which a move request was issued contains line operator history, *MOVE. The top right corner of the panel contains a short message, such as REQUEST ID:1932.

Each line operator field associated with an entry for which no move request was issued contains -MOVE.

- If you entered N on the Move Confirmation panel, the ISMF Minidisk List panel reappears, and no move requests are issued. Each line operator field in the list contains -MOVE. For information about entries that contain invalid information, refer to your ISPF log.

Note: Issue the CLEAR command before issuing another MOVE list request.

Notes About Moving Minidisks:

- Before you issue move requests, you might find it helpful to create and save lists that provide you with information that can help you make decisions related to moving minidisks.

For example, before you move minidisks among volumes to perform routine space utilization tasks, you can create a minidisk list that contains all the contiguous minidisks on a volume. You can then sort that list by minidisk gap.

The sorted list identifies any free and unused space between minidisks on a specific volume. You can then create the lists of minidisks to be moved to consolidate space on those volumes.

- You can also create lists that group minidisks together in appropriate subsets for your move operation. For example, when you migrate minidisks to new DASD, you can create and save manageable lists of minidisks to be moved. You can then submit your move requests at the most convenient time for your installation.
- Choosing an I/O buffer size allows you to speed up the move process by specifying a larger number of tracks or blocks to move at a time.

However, when you choose an I/O buffer size, you should consider the number of server virtual machines that your installation allows to perform move operations concurrently (specified in the DFSMS/VM control file), and the amount of total system memory you have available. Specifying a large I/O buffer size could affect your overall system performance.

Also, changing the I/O buffer size is effective only when DFSMS COPY is used.

- You can use the I/O buffer size option with the pacing level option to help control the impact of your move operation on system performance. You can choose a continuous or intermittent pacing level.

Verifying the Integrity of CMS Minidisks

You can issue a check request to perform an integrity evaluation of a CMS minidisk to verify whether its file structure is intact. The results are stored in a CMS file. You might want to issue a check request after moving a minidisk, or to check for integrity problems when you receive a problem report from a general user. The following sections provide step-by-step instructions to issue a check request.

Check Options

When you issue a check request, ISMF displays an entry panel that allows you to specify:

- Where to send the results (the virtual reader for your user ID or a different user ID)
- Whether to send any results or only error information
- The number of times to attempt to relink to a minidisk and the time interval between each attempt

Check Results

The results of a check request include:

- Total number of blocks
- Number of blocks used
- Used blocks counted
- Allocation map bits set
- Disk origin pointer
- Files reported in directory
- Number of files found

A check request also records any of the following types of abnormalities found:

- Damage to minidisk label
- Lost blocks
- Invalid blocks
- Overlapping blocks

If abnormal blocks are found, the check request also records the address of those blocks. If there are blocks that are used by more than one CMS file (overlapping blocks), all of the file names of the affected files are listed.

Issuing Check Requests

To check CMS minidisks:

1. Create or display the minidisk list that contains the CMS minidisks you want to check.
2. On the Minidisk List panel, type **CHECK** in the line operator field next to the entry for the first (or only) CMS minidisk you want to check as shown in [Figure 59 on page 107](#).

```
DGTLNML1
COMMAND ==>>

MINIDISK LIST PANEL

                                SCROLL ==>> HALF
                                Entries 1-14 of 16
                                Data Columns 4-7 of 13

ENTER LINE OPERATORS BELOW:
```

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	MDISK START	MDISK END	MDISK SIZE	MDISK GAP
--- (1) ---	-- (2) --	--- (3) ---	--- (4) ---	--- (5) ---	--- (6) ---	--- (7) ---
CHECK	MAINTDEV	1A1	30	129	100	0
	MAINTDEV	19F	130	229	100	0
	MAINTDEV	120	230	479	250	0
	MAINTDEV	19A	480	879	400	0
	MAINTDEV	BC1	880	909	30	496
	MONWRITE	191	1406	1605	200	0
	CSERVOX0	191	1606	1610	5	54
	RTMTSTA	191	1665	1694	30	0
	RTMTSTB	191	1695	1704	10	0
	RTMTSTC	191	1705	1714	10	0
	G32TST	191	1715	1744	30	165
	STUMP	191	1910	1929	20	0
	BORREG01	191	1930	1949	20	0
	BEACHMA	191	1950	1958	9	0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 59. Entering the CHECK Line Operator

If you are checking a single minidisk, go to step 3. If you are checking multiple minidisks, continue with the following steps.

To repeat the CHECK line operator:

Type an equal sign (=) next to any subsequent entries in the list that identify minidisks you want to check as shown in [Figure 60 on page 107](#).

```
DGTLNML1
COMMAND ==>>

MINIDISK LIST PANEL

                                SCROLL ==>> HALF
                                Entries 1-14 of 16
                                Data Columns 4-7 of 13

ENTER LINE OPERATORS BELOW:
```

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	MDISK START	MDISK END	MDISK SIZE	MDISK GAP
--- (1) ---	-- (2) --	--- (3) ---	--- (4) ---	--- (5) ---	--- (6) ---	--- (7) ---
CHECK	MAINTDEV	1A1	30	129	100	0
=	MAINTDEV	19F	130	229	100	0
=	MAINTDEV	120	230	479	250	0
=	MAINTDEV	19A	480	879	400	0
=	MAINTDEV	BC1	880	909	30	496
=	MONWRITE	191	1406	1605	200	0
=	CSERVOX0	191	1606	1610	5	54
=	RTMTSTA	191	1665	1694	30	0
=	RTMTSTB	191	1695	1704	10	0
=	RTMTSTC	191	1705	1714	10	0
=	G32TST	191	1715	1744	30	165
=	STUMP	191	1910	1929	20	0
=	BORREG01	191	1930	1949	20	0
=	BEACHMA	191	1950	1958	9	0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 60. Repeating the CHECK Line Operator

Repeat the CHECK line operator when you want to check more than one minidisk, and you want to use different options for each check request.

When you repeat the CHECK line operator, the Check Entry panel is displayed repeatedly for each check request.

Go to step 3.

To use the CHECK line operator in last-use mode:

- a. Type == (two equal signs) in the line operator field next to the second entry that identifies a minidisk you want to check.
- b. Type = in the line operator field for each subsequent entry that identifies a minidisk you want to check.

Figure 61 on page 108 shows sample entries to use the CHECK line operator in last-use mode.

DGTLM1

COMMAND ==>

MINIDISK LIST PANEL

SCROLL ==> HALF

Entries 1-14 of 16

Data Columns 4-7 of 13

ENTER LINE OPERATORS BELOW:

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	MDISK START	MDISK END	MDISK SIZE	MDISK GAP
--- (1) ---	-- (2) --	- (3) -	--- (4) ---	--- (5) ---	--- (6) ---	--- (7) ---
CHECK	MAINTDEV	1A1	30	129	100	0
==	MAINTDEV	19F	130	229	100	0
=	MAINTDEV	120	230	479	250	0
=	MAINTDEV	19A	480	879	400	0
=	MAINTDEV	BC1	880	909	30	496
=	MONWRITE	191	1406	1605	200	0
=	CSERVOX0	191	1606	1610	5	54
=	RTMTSTA	191	1665	1694	30	0
=	RTMTSTB	191	1695	1704	10	0
=	RTMTSTC	191	1705	1714	10	0
=	G32TST	191	1715	1744	30	165
=	STUMP	191	1910	1929	20	0
=	BORREG01	191	1930	1949	20	0
=	BEACHMA	191	1950	1958	9	0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 61. Using the CHECK Line Operator in Last-Use Mode

Use the CHECK line operator in last-use mode when you want to check more than one minidisk, and you want to use the same options for each check request. When you use last-use mode for the CHECK line operator, the Check Entry panel is displayed only once.

Go to step 3.

3. Press **ENTER**.

The Check Entry panel appears. Figure 62 on page 109 shows the defaults that appear on that panel. If you repeat the line operator or used last-use mode, the minidisk name and the output file name and file type are those associated with the first request you entered.

The output destination user ID shown is a sample default value, and the output destination system is a sample entry.

```

DGTDC01                                CHECK ENTRY PANEL
COMMAND ===>

MINIDISK:  MAINTDEV 1A1
FILENAME AND FILETYPE FOR RESULT:  MAINTDEV 01A1CHK

OPTIONALLY SPECIFY ONE OR MORE:

OUTPUT DESTINATION USERID  ===> SMITH      (user ID, 1-8 characters)
OUTPUT DESTINATION SYSTEM  ===> ABCVM1     (system ID, 1-8 characters)
OUTPUT RESULTS LEVEL       ===> 1          (1=any result, 2=fail only)
NUMBER OF RETRIES FOR LINK  ===> 2          (0 to 99)
LINK RETRY INTERVAL        HOURS ===> 00    MIN ===> 00    SEC ===> 10

USE ENTER TO CONTINUE.
USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

```

Figure 62. Check Entry Panel

4. Fill in the Check Entry panel with the options you want to use, or leave the default values as shown. You must enter a value in the OUTPUT DESTINATION SYSTEM field.

You can specify whether you want any results returned by a check request or only error information. You can also indicate the number of times to attempt to relink to a minidisk and the time interval between each attempt. Note that a link to a minidisk is established only if the server virtual machine processing your request can obtain an exclusive link. That is, there are no current read or write links to the minidisk.

When the Check Entry panel is displayed, press **PF1** to select help panels that contain detailed information about each field.

5. Press **ENTER**.

- If you issue one check request, the Minidisk List panel is displayed. The line operator field contains line operator history, such as, *CHECK. The top right corner of the panel contains a short message, such as CHECK ISSUED OK.
- If you repeat the CHECK line operator, the Check Entry panel appears for each request you issue. Fill in the panel for each request.

When you press ENTER after filling in the panels for the last request, the ISMF Minidisk List panel is displayed. The line operator fields for each request contain line operator history. The top right corner of the panel contains a short message related to the last request processed.

- If you used the CHECK line operator in last-use mode, the line operator fields for each request contain line operator history. The top right corner of the panel contains a short message related to the last request processed.

Notes About Check Output:

- You will receive a DFSMS/VM response file when the CHECK request completes. The file will contain messages that pertain to the check request. The following is an example file: [Figure 63 on page 109](#) shows sample results of a check request.

```

DFSMS Function Level 221                                04/08/05   03:22:03
DFSMS CHECK, request identifier 189

The following options were specified for the Source Minidisk:
  Output User: VMTEST0   Output System: VMDRV5   Results Level: 1
  Link Retries    2       Link Interval: 000010

FSMEMD4000I DFSMS CHECK Request identifier 189 (DFSMS 299) succeeded

DFSMS CHECK completed

```

Figure 63. Sample Check Request Results

- Check output is stored in a file sent to the virtual reader associated with the user ID you specify. [Figure 64 on page 110](#) shows sample results of a check request.

```

Userid: DFSMS      Date: 04/08/05   Time: 05:37:06
CHECK  299  (
Address: 0299  Device Type: FB-512  Date Created: 2005/02/19 16:26:38
Valid: MY299  Block size: 4096  Last Changed: 2005/02/19 16:27:10
FBA blocks:      Total:      64 Usable:      64

Total number of CMS blocks:      8
Number of CMS blocks used:      6  ( 75%)
CMS blocks counted:      6
Blocks in allocation map:      6

Lost CMS blocks:      0
Invalid CMS blocks:      0
Overlapping CMS blocks:      0

Disk origin pointer:      5
Files reported in directory:      3 (Including DIRECTOR and ALLOCMAP)
Number of files found:      3

Command CHECK  gave return code =  0

```

Figure 64. Sample Check Output

When the check output screen is displayed, press **PF1** to select help panels that contain detailed information about each field in the check output file.

- CHECK can detect minidisk errors from problems caused by file system errors, errors in vendor or IBM products, and hardware errors.

If CHECK detects an error, report it to your system programmer, who should do the following:

1. Look for a pattern in reported errors that indicates a common problem in hardware or software. For example, general users encountering problems might be using the same program or vendor product or have minidisks on the same volume.
2. Call the IBM Support Center if you believe that the problem is caused by the CMS file system.

Request IDs

When ISMF accepts MOVE or CHECK requests, it records a request ID for the request in a hidden field. You need to know the current request ID when you issue a QUERY or DISCARD command. The QUERY or DISCARD line operator, however, does not require you to enter a request ID.

Be sure to obtain the request ID of any request you plan to track by using the MESSAGE or QUERY command *before* you issue another request against the same entry. ISMF updates the request ID field each time that you enter a new request against a list entry.

You can obtain the request ID by displaying a long message associated with the request in one of the following ways:

For the last request accepted (a short message, such as MOVE ISSUED OK, appears at the top of the screen):

- Enter **HELP** or press **PF1**. A long message appears that includes the request ID; for example:

```
MOVE ISSUED SUCCESSFULLY WITH REQUEST ID: 105
```

For a specific request:

1. Enter **MESSAGE** in the line operator field next to the entry for which you want to obtain the request ID. A short message, such as CHECK ISSUED OK, appears at the top of the screen.
2. Press **PF1** to display a long message that includes the request ID.

The group request ID associated with a command line move is identified in a short message, for example, REQUEST ID: 1614.

Chapter 11. Starting and Stopping DFSMS/VM Processing

This section describes initialization and termination of the DFSMS/VM master and server machines. These machines consist of the DFSMS master, DFSMS server, and the minidisk server machines.

Starting DFSMS/VM

You have several options for starting (also known as—initializing, activating, or logging on) the master and server virtual machines. The start-up procedure is either initiated automatically or manually—automatically is the recommended method.

Automatic logon involves logging on the virtual machines using either the AUTOLOG1 or AUTOLOG2 virtual machine. Manual logon involves issuing the XAUTOLOG or AUTOLOG command from an authorized user ID. Refer to the *Program Directory for function level 221* for details about the DFSMS/VM logon process.

During initialization, the control file (DGTVCNTL DATA located in the SFS directory VMSYS:DFSMS.CONTROL) is read and access to required directories and file pools is verified. The system is ready to accept requests for system processing when initialization of the master machines and server machines is complete. If errors occur, DFSMS/VM logs messages to the console identifying the problem area and then stops. Refer to the *z/VM: DFSMS/VM Messages and Codes* for information about DFSMS/VM messages. There are some situations that may slow the DFSMS initialization process. An example of this is when either the VMSYS file pool or the work directory is not available. DFSMS will wait a certain amount of time and if any of these items do not become available within an appropriate period of time, an error message is logged and DFSMS/VM stops.

Stopping the DFSMS Master and Server Machines

DFSMS should be running as much as possible so that users with migrated files can have access to their migrated files. Reasons for stopping DFSMS include:

- Servicing the product
- Tailoring the installation-wide exits
- Tailoring the DFSMS control file (control file changes are not recognized by DFSMS until it is stopped and restarted)
- If RACF/VM is being used for security and DFSMS loses communication with RACF/VM

An authorized user can stop the DFSMS master and server machines by issuing the DFSMS STOP SMS command or the ISMF STOP command. The following are the two options for the STOP command:

- The QUIESCE option allows work that has been accepted by the DFSMS master to complete before shutdown. QUIESCE is the default.
- The IMMEDIATE option only allows work that is active, except for work begun by a MANAGE or CONVERT command, to complete.

Work that was accepted is discarded and restarted when DFSMS/VM is reinitialized, except for MANAGE or CONVERT. A MANAGE or CONVERT request that is executing or accepted is discarded and not restarted. Be aware that once a stop is initiated, it cannot be cancelled and that no requests are accepted while DFSMS/VM is stopping.

The DFSMS master machine also stops if a critical task (or process) terminates due to errors. When the DFSMS master stops, it signals each DFSMS server and minidisk server to stop. Therefore, the DFSMS master and its servers all stop when the DFSMS master stops; the only way to stop a server is by having the DFSMS master signal the server to stop. For additional information about the DFSMS STOP

SMS command, see [“DFSMS STOP”](#) on page 161. And for additional information about the ISMF STOP command, see [“Using ISMF Commands and Line Operators”](#) on page 162.

STOP QUIESCE

After an authorized user issues a DFSMS STOP SMS (QUIESCE or STOP QUIESCE from within ISMF command, DFSMS only accepts the following commands:

- DFSMS DISCARD
- DFSMS QUERY
- DFSMS STOP SMS (IMMEDIATE
- ISMF DISCARD
- ISMF QUERY
- ISMF STOP IMMEDIATE

Work requests that are processing and waiting in the queue, when this command is issued, will complete processing before DFSMS/VM stops.

STOP IMMEDIATE

When a DFSMS STOP SMS (IMMEDIATE or a STOP IMMEDIATE from within ISMF command is issued, DFSMS/VM begins the shut-down process and does not accept any more work requests. The work in process is completed with the exception of MANAGE or CONVERT requests. DFSMS CONVERT and MANAGE command processing is discarded. The following is processing that is stopped and restarted when DFSMS/VM is reinitialized:

- DFSMS ALTER (NOWAIT
- DFSMS MIGRATE (NOWAIT
- DFSMS RECALL (NOWAIT
- Operations on minidisk lists

EXAMPLE:

Commands issued with pattern-matching characters can process multiple files. For example, if there are ten files with the file type of EXEC and you issue:

```
DFSMS MIGRATE * EXEC A (NOWAIT
```

all ten files with the EXEC file type are eligible to be migrated. Since only one file can be processed by a task in an SMS server, files must wait to be processed by an available task.

If the DFSMS STOP SMS (IMMEDIATE is issued while processing, migrate processing stops with any files that are waiting to be processed, but finishes processing once DFSMS/VM is reinitialized.

Chapter 12. User Authorization and Security Controls

The need for data security is increasing as more people become familiar with data processing, computers become easier to use, and more critical assets are stored on computers. This chapter provides information regarding security when using the DFSMS/VM product. Refer to the [z/VM: DFSMS/VM Customization](#) for details about techniques for authorizing the use of DFSMS/VM.

DFSMS/VM authorization processing does not override SFS authorization controls.

Security Controls

You control access to DFSMS/VM functions in one of four ways:

- Use the RACF/VM program product, which allows you to restrict the use of DFSMS commands.
- Use the simple authorization scheme of listing users authorized to perform storage administrator functions in an authorization file.
- Use a different facility for security, other than RACF/VM, that uses the RACROUTE interface.
- Use an installation-wide exit to replace or enhance the previous methods. This method can also be used to override all other methods.

Using RACF/VM for Security

The RACF[®] licensed program is IBM's primary tool for managing security. In the z/VM environment, RACF/VM is used to identify and verify a user's authority to both access data and to use system facilities. DFSMS/VM uses the FACILITY class of RACF/VM for command-level protection. A single generic profile (STGADMIN.*) that protects all DFSMS commands can be created since the command entities begin with either STGADMIN (for DFSMS/VM storage management commands) or STGADMIN.RM (for DFSMS/VM removable media services commands). If you need more flexibility, you can also create discrete profiles to specify authorization command-by-command, user-by-user. For additional information about discrete profiles, refer to the [z/VM: DFSMS/VM Customization](#). For additional information about using RACF, refer to the *RACF Security Administrator's Guide*.

DFSMS/VM Authorization File

Anytime that RACF/VM or a equivalent product is not available or not installed, DFSMS/VM uses the authorization file (DGTVAUTH DATA in the SFS directory VMSYS:DFSMS.CONTROL) to determine authorization. For example, if communication between DFSMS/VM and RACF is lost, DFSMS/VM automatically defaults to the authorization file.

Note: You must stop and restart DFSMS/VM to reconnect with RACF/VM.

When using the authorization file for authority checking, DFSMS/VM differentiates between general user commands and storage administrator commands. The authorization file contains the user IDs that have access to the storage administration commands. There are five DFSMS commands that the general user can invoke:

- DFSMS DISCARD
- DFSMS MIGRATE
- DFSMS QUERY DFSMSLEVEL
- DFSMS QUERY REQUEST
- DFSMS RECALL
- DFSMS REPORT SPACEMANAGEMENT FILESPACE

A user with storage administrator authority can also invoke these five commands plus all other commands used in DFSMS/VM.

Using a Different Facility for Security

You can use a security facility other than RACF/VM. If you choose a security facility that uses the RACROUTE interface, as RACF/VM does, you simply install that program instead of RACF/VM. DFSMS/VM uses whatever program answers to the RACROUTE interface. For additional information about this interface, refer to the [z/VM: Security Server RACROUTE Macro Reference](#).

Using an Installation-Wide Exit

If you choose a security facility that does not use the RACROUTE interface or if you choose to devise your own security method, you must customize the authorization process. Customizing is done by replacing the CSL routine, FSMVAUTH. For information about installation-wide exits, see [z/VM: DFSMS/VM Customization](#).

The DFSMS Command Authorization Process

The actual method of authorization varies from installation to installation. [Figure 65 on page 115](#) shows the default DFSMS/VM authorization process.

Note: When the FSMVAUTH installation exit return code equals four, DFSMS/VM authorization processing is activated.

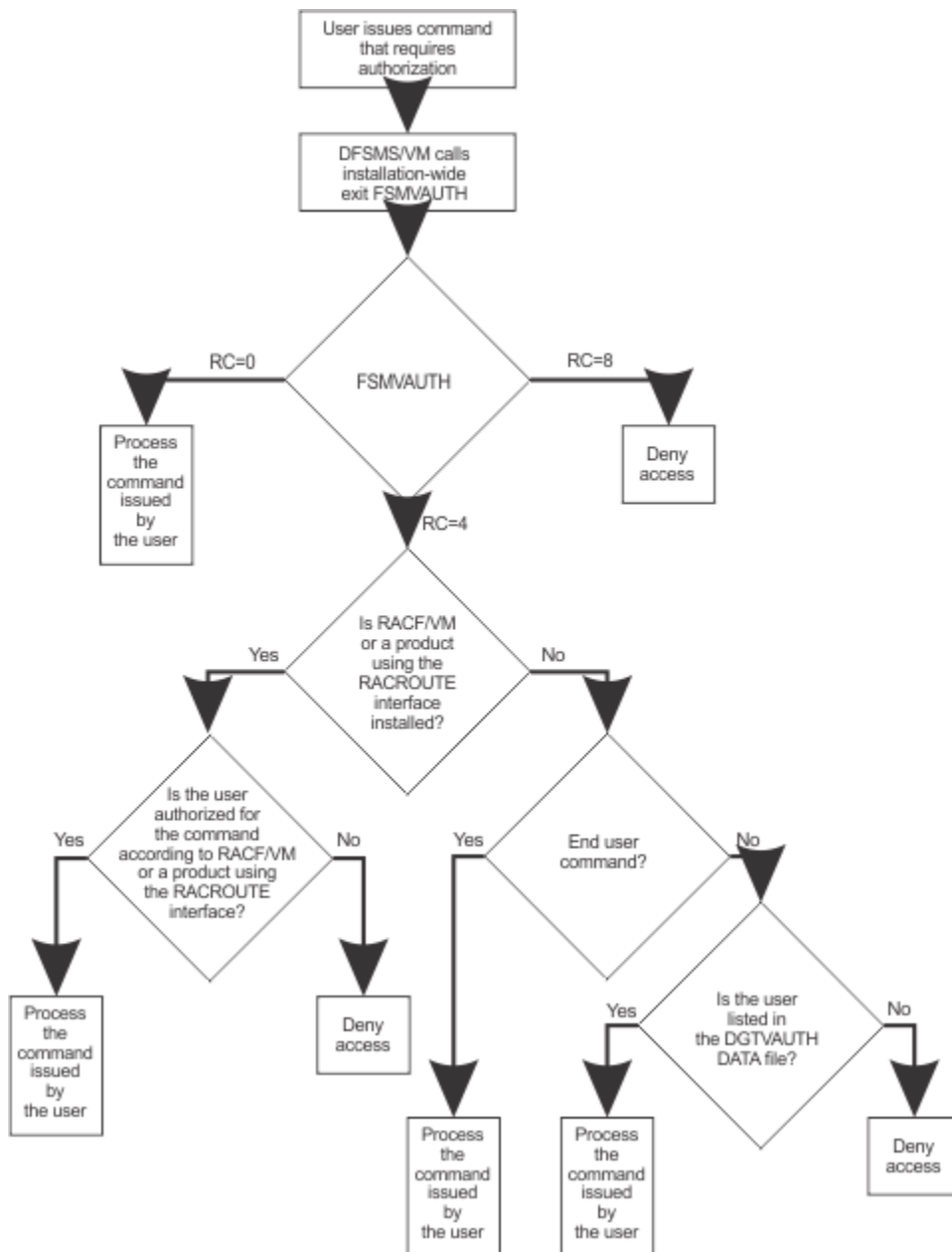


Figure 65. The DFSMS Command Authorization Process

Security of File Spaces That DFSMS/VM Uses

DFSMS/VM stores migrated data, logging information, and its control information in SFS files. You need to ensure that these files are not accessed nor updated by unauthorized users. Since these are SFS files, you are able to ensure file security by using normal SFS authorization techniques. You need to ensure the security of the following:

1. Files that contain migrated data (ML1 data). These files are stored in the directory structure indicated by the MIGRATION_LEVEL_1 control file parameter.

No one should modify these files nor the directory structure, except when naming convention changes are required. See [“Renaming Migration Level 1 and Subdirectory Names”](#) on page 76 for instructions on renaming the secondary storage structure.

2. Files that contain DFSMS/VM log information. If the installation specified that DFSMS/VM log information to a file via the DFSMS_LOG_TO_FILE control file parameter, this information is written to SFS files. The installation is free to modify or erase any of these files, except the installation should

ensure that no modifications are made to the current day's files since DFSMS/VM may be writing to them.

3. Files stored in the directory indicated by the WORK_DIRECTORY control file parameter. These files are used exclusively by DFSMS/VM, and the installation must ensure that no modification of these files occurs.
4. The directory structure under VMSYS:DFSMS.
5. The DFSMS/VM product code and all VMSES/E disks.
6. Alteration of the shared segment can only be performed by DFSMS/VM.
7. If the ML2 capability is being used, the data on the ML2 server must be protected. See "Establish TSM Registration and Authentication" in [*z/VM: DFSMS/VM Customization*](#).

Other Security Considerations

That general users have read access to a RACF entity for a command does not necessarily mean that they can invoke the command. For example, in order to migrate a file, a general user must have WRITE authorization to that file; in order to recall a file, a general user must have READ authorization to that file. If a general user submits a DFSMS RECALL or MIGRATE command using pattern matching characters, DFSMS/VM will only attempt to process the files for which the general user is sufficiently authorized.

However, to allow a non-SFS administrator (for example, a system operator) to run the DFSMS MANAGE, DFSMS CONVERT, or DFSMS ALTER commands, only storage administrator authority is required in order to issue these commands.

Chapter 13. Managing Storage Using DFSMS/VM

DFSMS/VM storage management consists of various actions performed by you and the system. The following is information provided in this section:

- Managing SFS storage groups
- Migrating files to secondary storage
- Recalling files from secondary storage
- Converting SFS storage groups
- Reporting on migrated data

Secondary storage, consisting of migration level 1 (ML1) and migration level 2 (ML2), is the location where DFSMS/VM places migrated files. If additional information is required, see [Chapter 7, “Specifying DFSMS/VM Secondary Storage,”](#) on page 75.

In the following sections, the management class used to manage a file is referred to as a file’s management class, whether the management class used is actually obtained from the file, the parent directory, or the system default management class. For example, when a file’s management class is NULL, the parent directories’ management class is used to manage the file. For information concerning management classes, see [“Management Class”](#) on page 7 and [Chapter 4, “Management Classes,”](#) on page 39.

Information that will broaden your knowledge of how management classes, configurations, and ACS processing work together is available in [Appendix H, “Implementing Configurations,”](#) on page 249.

Managing a Storage Group

The DFSMS MANAGE command erases and migrates eligible files from an SFS storage group according to the parameters in each file’s management class. The storage group specified in the command can be any storage group except storage group 1. The DFSMS MANAGE command also moves files from ML1 to ML2. The file movement is determined by the management class attributes. If a file’s management class indicates migration to ML2, but ML2 is not specified in the control file, the file is migrated to ML1.

Performance tip: Use the DFSMS MANAGE command during low system activity. This results in less impact to performance during prime operating hours.

The LEVELS parameter can give a greater level of flexibility and precision to the DFSMS MANAGE command. For example, if a storage group is running out of space during prime shift, a MANAGE command with LEVELS ML1 may be appropriate to quickly move files to DASD when there is not time to do tape mounts.

Another DFSMS MANAGE parameter, WAIT/NOWAIT gives the flexibility of issuing a group of MANAGE commands simultaneously. The MANAGE commands can be placed into an EXEC with the WAIT option which allows them to be processed one by one.

During DFSMS MANAGE processing when using the THRESHOLD option, DFSMS checks the space utilization of the storage group and compares it with the threshold values specified in the control file (thresholds are expressed as percentages). The control file statements that identify the thresholds for all storage groups on the system are STORGROUP_LOW_THRESHOLD and STORGROUP_HIGH_THRESHOLD.

The following conditions describe how these control file parameters are used to tailor MANAGE command processing:

- If storage group utilization is below the specified high threshold, erasure, migration, or file movement between ML1 and ML2 is not performed.
- If storage group utilization is equal to or greater than the specified high threshold, erasure of all expired files occurs first. If the storage group utilization is still above low threshold, then eligible files are migrated one-by-one until low threshold is reached or there are no more files eligible for migration.

During DFSMS MANAGE command processing, if a file or directory is found with no management class, ACS processing is run on the file or directory before space management processing continues.

A file is erased during DFSMS MANAGE processing if the following conditions are met:

Note: *Erased* means either the data and file structure are deleted or only the data is deleted (see [“Expiration Attributes” on page 39](#)).

- The management class is in the active configuration.
- The expiration criteria is satisfied (see [“Expiration Attributes” on page 39](#)).
- The installation-wide exit FSMMECHK allows it.
- The first five characters of the top-level directory name (or file space) do not begin with DFSMS.
- The storage is above high threshold if the THRESHOLD option is specified.
- The file cannot be in a directory control directory that is accessed read/write by someone else.

A base file is migrated during DFSMS MANAGE command processing if the following conditions are met:

- The management class is in the active configuration.
- The migration criteria is satisfied (see [“Migration Attributes” on page 40](#)).
- The installation-wide exit FSMMECHK allows it.
- The first five characters of the top-level directory name (or file space) do not begin with DFSMS.
- The file is not under directory control (in a dircontrol directory).
- The file is not empty.
- The file’s date of last reference is not zero.
- The storage is above low threshold if the THRESHOLD option is specified.

Note: Date of last reference (DOLR) is a file attribute used to determine when a file is migrated or expired. This date is not immediately changed when the file is referenced. Files with a DOLR of zero are not erased or migrated by the DFSMS MANAGE command. Files with a zero date of creation are not erased by the MANAGE command. See the [z/VM: CMS Application Development Guide](#) for additional information about DOLR and date of creation.

The following are ML2 considerations during DFSMS MANAGE command processing:

- A file is eligible for migration from primary storage directly to ML2 if the SECONDARY STORAGE NON-USAGE management class parameter is set to zero days.
- A file is eligible to move from ML1 to ML2, when the secondary storage non-usage criteria is satisfied.

During DFSMS MANAGE command processing, if the management class is not found in the active configuration, the file is not processed and an error is issued. There are several ways to correct this type of error. For example, the DFSMS ALTER (or ISMF ALTER) command can be used to assign the file a valid management class. Also, the DFSMS ACTIVATE (or ISMF ACTIVATE) command can be issued to activate a configuration that defines the previously undefined management class. Or if a management class is no longer required, ACS processing can be modified so that a valid management class is associated with those files. Then by running DFSMS CONVERT with the REDETERMINE option, the files are then associated with the valid management class.

Managing a storage group is an asynchronous activity. Unless the WAIT option is used, your virtual machine regains control upon acceptance of the request, not completion. When the command completes, a report is sent to your reader. This report contains a unique request id assigned by DFSMS that can be used to reference messages in log files. The report also contains messages which are displayed for each erased file, for each file or directory processed through ACS, and for each file not managed due to the FSMMECHK exit. The summary of the processing performed includes the number of migrated or erased files and 4K blocks.

Note: Commands that can assist you when working with the DFSMS MANAGE command are the DFSMS or ISMF DISCARD and QUERY commands. QUERY displays active and queued requests. DISCARD stops processing of manage requests. Refer to [Chapter 14, “DFSMS Commands,” on page 131](#) for descriptions

of the DFSMS commands, and “Using ISMF Commands and Line Operators” on page 162 for descriptions of the ISMF commands.

The syntax of the DFSMS MANAGE command is illustrated in “DFSMS MANAGE” on page 147. An example DFSMS MANAGE command report format is shown in Figure 66 on page 119.

```
DFSMS Function Level 221                                02/30/2001 09:02:41
DFSMS MANAGE with NOWAIT, request identifier 120

DFSMS MANAGE processing started for storage group 32767 in file pool VMSYSU

1
FSMDSM3141I File STATUS LIST3820 VMSYSU:BELL.TEST. was erased
FSMDSM3141I File WEEK1 MODULE VMSYSU:BELL.TEST. was erased
FSMDSM3136I Migration of file TESTA1 EXEC VMSYSU:BELL.TEST. not allowed by FSMMECHK exit,

DFSMS MANAGE STORGRP command summary:

2 High Threshold: 80 Low Threshold: 30
3 Storage group percent utilization before MANAGE: 95
4 Storage group percent utilization after MANAGE: 50
5 48110 blocks available in primary storage group
5 24953 blocks available in ML1
6 0 files and directories converted
7 0 files and directories could not be converted
8 2 files occupying 2 4K blocks erased
9 50 files eligible to be migrated
10 43 files occupying 3000 4K blocks have been migrated from primary storage to ML1
10 7 files occupying 276 4K blocks have been migrated from primary storage to ML2
11 24 files occupying 1478 4K blocks have been moved from ML1 to ML2

DFSMS MANAGE with NOWAIT completed with no errors
```

Figure 66. Example of the DFSMS MANAGE Command Output

The report primarily contains two areas:

- The area denoted by **1** contains messages indicating any errors and files that have been erased or converted.
- The area denoted by **2** through **11** contains the command summary. These messages are always displayed in the report. The following is a brief description of the messages in the command summary (for a detailed description of these messages, see [z/VM: DFSMS/VM Messages and Codes](#)):

2 FSM3147—This line is only displayed when the DFSMS MANAGE command’s THRESHOLD option is issued. The displayed information indicates the high and low thresholds for the processed SFS storage group. These threshold values are specified and can be changed in the DFSMS/VM control file (DGTVCNTL DATA).

3 FSM3148—This line displays the percentage of the storage in use by SFS prior to DFSMS MANAGE command processing.

4 FSM3149—This line displays the percentage of the specified storage group SFS is utilizing after DFSMS MANAGE command processing.

5 FSM3106—These lines display the number of available blocks when DFSMS MANAGE command processing completes.

6 FSM3174—This line displays the number of files and directories that have been assigned a management class during DFSMS MANAGE command processing.

7 FSM3176—This line displays the number of files and directories without a management class that could not be assigned a management class during DFSMS MANAGE command processing.

8 FSM3145—This line displays the number of files and 4K blocks erased during DFSMS MANAGE command processing.

- 9** FSM3105—This line displays the number of files eligible for migration, according to the level and threshold parameters specified in the management class attributes.
- 10** FSM3145—These lines display the number of files and 4K blocks of primary storage migrated to either ML1 or ML2 during DFSMS MANAGE command processing.
- 11** FSM3145—This line displays the number of files and 4K blocks of ML1 storage moved from ML1 to ML2 during DFSMS MANAGE command processing.

Exit before Migration or Expiration of a File

The installation-wide exit FSMMECHK is available during file expiration or migration. This exit optionally allows the installation to prevent file migration or erasure from occurring. For additional information about the FSMMECHK exit, refer to [z/VM: DFSMS/VM Customization](#).

The MIGRATE Command

The DFSMS MIGRATE command migrates files from primary storage to secondary storage (ML1 and ML2). If a file has no management class, ACS processing is invoked and a management class is assigned. During migration processing the management class has two roles (1) it indicates whether a primary file is eligible for migration and (2) if a file is eligible for migration, it indicates whether a file should migrate to ML1 or bypass ML1 and migrate directly to ML2. If a file's management class indicates migration to ML2, but ML2 is not specified in the control file, the file is migrated to ML1. The two management class parameters indicating these two roles are *SPACE MANAGEMENT TECHNIQUE* and *SECONDARY STORAGE NON-USAGE* respectively. After migration processing completes, migrated files are identified in the file pool catalog entries. DFSMS MIGRATE does not move files from ML1 to ML2; only DFSMS MANAGE has this capability. Only base files or aliases can be specified with the DFSMS MIGRATE command. If a file specified is an alias, the base file associated with the alias is processed. Pattern-matching characters are allowed for the file name and file type. This allows designation of a group of files (including the base files for any aliases that are matched) rather than processing individual files. For additional information, see [“Pattern Matching”](#) on page 134.

To migrate a file using the DFSMS MIGRATE command:

- The requester must have write authority to the file. Any file for which write authority has not been granted is not migrated.
- The management class associated with the file must have the value MIGRATE in the SPACE MANAGEMENT TECHNIQUE field.
- The file pool where the file resides is DFSMS/VM managed and is up and operational. If not, the file is not managed and an error is logged.
- The first five characters of the top-level directory (owner user ID) cannot be DFSMS.
- The file can not be in a directory under directory control.
- The base file can not be a empty file.
- The file can not reside in the same file pool and storage group as the ML1 file space for this VM system.
- The file's date of last reference can not be zero.

When the DFSMS MIGRATE command completes processing, a report is sent to your reader. The report contains the request ID, error messages for any files not migrated, and a summary of the processing performed. When a MIGRATE request is submitted with no pattern-matching character (* or %) specified with the file name or file type, it is assumed that a single file is to be migrated. In this case, if the MIGRATE is not successful, a specific error message will be issued.

Otherwise, it is assumed one or more files can be selected for migration. In this case, no error message is given when a file does not qualify for migration.

Note: Commands that can assist you when working with the DFSMS MIGRATE command are the DFSMS DISCARD and QUERY commands. QUERY displays active and queued requests. DISCARD stops processing of migrate requests. RECALL processing, whether from DFSMS RECALL command or from an automatic RECALL command from SFS recalls migrated files in ML1 or ML2. Refer to [Chapter 14, “DFSMS](#)

Commands,” on page 131 for descriptions of the DFSMS commands. The syntax of the DFSMS MIGRATE command is illustrated in Chapter 14, “DFSMS Commands,” on page 131 and a sample report is shown in Figure 67 on page 121.

```
DFSMS Function Level 221                                02/16/2001  16:50:39
DFSMS MIGRATE with NOWAIT, request identifier 57

DFSMS MIGRATE processing started for WEEKLY * VMSYSU:USER01.

 1      7 files matched the fileid specified
 2      6 files were eligible for processing
 3      2 files migrated to migration level 1
 4      4 files migrated to migration level 2

DFSMS MIGRATE with NOWAIT completed with no errors
```

Figure 67. Example of the DFSMS MIGRATE Command Output

The following is a brief description of the standard messages displayed in the DFSMS MIGRATE command output (for a detailed description of these messages, see [z/VM: DFSMS/VM Messages and Codes](#)).

- 1** FSM3211—This line indicates that a pattern-matching character (*) has been specified for the file type, and that seven files with the file name of WEEKLY match the migrate file ID criteria.
- 2** FSM3212—This line indicates that six files are eligible, meaning that one of the files that match the file ID criteria does not meet processing criteria (for example, one of the files is a previously migrated file).
- 3** FSM3132—This line indicates the number of files migrated to ML1. A file can be eligible for processing as indicated in the last field, but may not be migrated. Two reasons for no migration are that the file is locked by a general user or the management class indicates that the file is not to be migrated.
- 4** FSM1207—This line indicates the number of files migrated to ML2.

Recall

Recall is the process of moving your file from secondary storage back to primary storage. The recalled file is returned to and stored in its original SFS file pool and the file attributes are updated to show that the file is no longer migrated. ACS is invoked for the recalled file and if needed a new management class could be assigned by ACS. If an error in ACS processing occurs, the file is still recalled, but no management class is assigned to the file. The installation can customize ACS processing (intentionally or unintentionally) to prevent a file from being recalled. Exercise care when customizing ACS (see Chapter 6, “Automatic Class Selection,” on page 63 for more details).

Recall can be initiated either:

- Automatically, by referring to a migrated file
- Manually, by issuing a DFSMS RECALL command

Automatic Recall

Whenever a migrated file’s data is opened, copied, browsed, or edited, SFS uses DFSMS/VM to recall the file. If you reference an alias to a base file that is migrated, the base file is recalled. Data is not recalled for operations that access data; for example, RENAME data is not recalled.

For performance reasons, there are times when a file is not recalled; instead it is erased. A file is erased rather than recalled when:

- You erase a migrated file.

- You indicate you want to replace all file data in a migrated file (for example, DMSOPEN with the REPLACE option or the COPYFILE REPLACE option). In both cases, SFS notifies DFSMS so it can erase the migrated copy of the file.

Automatic recalls from ML2 can cause tape mounts. Automatic recalls can be avoided if the DFSMS REPORT SPACE MANAGEMENT command is used to see if there are files in ML1 and ML2. Based on the report you receive from DFSMS, you can then decide how to set the CMS SET RECALL ON|OFF command. This command stops any automatic recalls of migrated data from either ML1 or ML2. See the CMS SET RECALL ON|OFF command in the *z/VM: CMS Commands and Utilities Reference* for more information. You can still get your files back by using DFSMS RECALL even if the CMS SET RECALL is off.

Recall by Command

If you know that a migrated file is required, you can issue a command directly to DFSMS/VM to recall the file.

To determine which files are migrated, and those that you need to recall, use any of the following tools to assist you:

- The CMS LISTFILE command with the SHARE option
- The CMS FILELIST command with the SHARE option
- The ISMF File Application
- The DFSMS REPORT SPACE MANAGEMENT FILESPACE command

The files specified in the DFSMS RECALL command are either base files or aliases. If a file specified is an alias, the base file associated with the alias is recalled. Pattern-matching characters can be used in the file name and file type, but not the directory. If pattern-matching characters are used, you must have at least read authority to the directory. If the recall is issued with pattern-matching characters, the unsuccessful recall of a file has no impact on subsequent file recalls.

To recall a file:

- You need at least read authority to the base file
- The file pool the file resides in is available to DFSMS/VM. If the file pool is not available, the file is not recalled, and an error is logged for the file
- ACS processing does not reject the file recall request
- There is enough storage group space to contain the recalled file

When the DFSMS RECALL command completes processing, a report is sent to your reader (see [Figure 68 on page 122](#)). The report contains the request ID that generated the command, error messages for any files that have not been recalled, and a summary of the processing performed. The MIGRATE and command RECALL can be discarded and queried. The syntax of the DFSMS RECALL command is illustrated in [Chapter 14, “DFSMS Commands,” on page 131](#).

```
DFSMS Function Level 221                12/01/2001   10:43:31
DFSMS RECALL with NOWAIT, request identifier 12

DFSMS RECALL processing started for TEST* ML1 VMSYSU:VMTEST0.

      3 files matched the fileid specified
      3 files were eligible for processing
      3 files recalled from migration level 1
      0 files recalled from migration level 2

DFSMS RECALL with NOWAIT completed with no errors
```

Figure 68. Example of the DFSMS RECALL Command Output

Converting a Storage Group

The DFSMS CONVERT command assigns management classes to files and directories in a storage group. The storage group specified in the command is any storage group except storage group 1.

Performance tip: Use the DFSMS CONVERT command during low system activity. This results in less impact to performance during prime operating hours.

When running the DFSMS CONVERT command, the default (NOREDETERMINE option) assigns management classes to files and directories with no management class. However, even if management classes are already assigned to files and directories, a new management class can be assigned to these files and directories using the DFSMS CONVERT command with the REDETERMINE option. This is required when changes are made to ACS processing or the configuration that conflict with management classes assigned to existing files and directories. (For example, if an existing management class named TAXES is not supported in a newly activated configuration, the files and directories that are assigned the TAXES management class need conversion.)

Consider running the DFSMS CONVERT command after you first install DFSMS/VM and want to assign management classes to files and directories that were created prior to DFSMS/VM installation. You can also wait until you run the MANAGE command, which also assigns management classes to files and directories that have no management class.

When the DFSMS CONVERT command completes processing, a report file is returned to your reader. This report contains the request ID, messages describing the success or failure of the assignment of management classes to the files and directories, and a summary of the processing performed.

The DFSMS QUERY or ISMF QUERY commands display active or queued convert requests. The DFSMS DISCARD or ISMF DISCARD commands discard convert requests. For descriptions of the DFSMS commands, see Chapter 14, “DFSMS Commands,” on page 131 and for descriptions of the ISMF commands, see [“Using ISMF Commands and Line Operators” on page 162](#). The report file indicates which files have changed.

The syntax of the DFSMS CONVERT command and a sample report is shown and described in [“DFSMS CONVERT” on page 140](#).

Reporting on DFSMS/VM Migrated Data

The DFSMS REPORT command provides information about migrated files, storage utilization, and out-of-sync conditions. The command generated report is for either a file space or storage group. The type of report is determined by your command keyword selection. The DFSMS REPORT SPACEMANAGEMENT FILESPACE results in a file space report and the DFSMS REPORT SPACEMANAGEMENT STORGROUP results in a storage group report. To run the DFSMS REPORT or DFSMS REPORT STORGRP commands for a general user other than yourself, you must have both SFS administrator and DFSMS storage administrator authority.

Example of Reporting on a File Space

Figure 69 on page 124 shows an example of the output, returned in a reader file, from the DFSMS REPORT SPACEMANAGEMENT FILESPACE command. The reader file returned upon completion of this command provides you with general information regarding the user’s file space.

```

DFSMS Function Level 221                                01/05/2001   14:57:20
DFSMS REPORT SPACEMANAGEMENT FILESPACE, request identifier 126

DFSMS REPORT SPACEMANAGEMENT FILESPACE processing started for file pool VMSYSU
and userid VMTEST2

```

```

1 Files migrated for this file space:                                5
  Migration Level 1:                                                2
  Migration Level 2:                                                3

2 Logical 4K blocks currently in use by this file space:            104

3 Physical 4K blocks in use by this file space:                      31

4 Physical 4K blocks in use by primary storage for this file space:  26

5 Physical 4K blocks in use by secondary storage for this file space: 5
  Migration Level 1:                                                3
  Migration Level 2:                                                2

6 Physical 4K blocks saved for this file space due to compaction:    75

```

The following files are migrated:

Filename	Filetype	Primary 4K Blocks	Management Class	Date Last Referenced	Date Migrated	Migration Level
Directory VMSYSU:VMTEST2.						
TEST1	DATA	5	DEFAULT	10/30/2001	10/29/2001	1
TEST1	ML1	50	ML1	10/30/2001	10/30/2001	1
TEST10	DATA	48	DEFAULT	10/30/2001	10/30/2001	1
TEST11	DATA	7	DEFAULT	10/30/2001	10/30/2001	1
TEST12	DATA	3	DEFAULT	10/30/2001	10/30/2001	1

DFSMS REPORT SPACEMANAGEMENT FILESPACE completed with no errors

Figure 69. Example of the DFSMS REPORT SPACEMANAGEMENT FILESPACE Command Output

The following is a brief description of the standard messages displayed in the DFSMS REPORT SPACEMANAGEMENT FILESPACE command output (for a detailed description of these messages, see [z/VM: DFSMS/VM Messages and Codes](#)).

- **1** FSM3126—Total number of migrated files for this file space. FSM3163 and FSM3164 display information separating the number of migrated files located in migration level 1 and migration level 2.
- **2** FSM3129—Logical 4K blocks in use by this file space. Indicating the total number of 4K blocks used by this file space as if no files are migrated. The number displayed here should match the resulting number if you issued a QUERY LIMITS command for this user. (That is, these are the committed blocks for this user.)
- **3** FSM3128—Physical 4K blocks in use by this file space. This number indicates the total number of physical blocks used by this file space. This number includes the space that migrated files are using in secondary storage in addition to the space that nonmigrated files are using in primary storage.
- **4** FSM3165—Physical 4K blocks in use by primary storage for this file space. It contains files that are not migrated.
- **5** FSM3127—Physical 4K blocks in use by secondary storage for this file space. This number is the total space used by migrated files for this file space in secondary storage. Since files are compacted in secondary storage, it should take less room to store the file in secondary storage than it would if this file remained in primary storage. For example, a file that takes up nine 4K blocks in primary storage may only take up three 4K blocks in secondary storage. This is dependent upon the data itself and the ability of DFSMS/VM to compress it. FSM3163 and FSM3164 display information separating the number of 4K blocks used in migration level 1 and migration level 2.
- **6** FSM3166—Number of physical 4K blocks saved due to file compaction.

Note: One 4K block is the minimum file compaction size; therefore, there is no space savings when single 4K block files are migrated. It is, however, advantageous to migrate single 4K block files when high-speed expensive DASD is used for primary storage and lower cost DASD is used for ML1 and tape for ML2.

In addition to the summary information, the report file may also show some error conditions such as:

- Files that are not migrated, but still have migrated data associated with them. These files are listed along with a delete token (see Figure 70 on page 125).

```
The following files are not migrated but DFSMS has migrated data for them:
```

Filename	Filetype	Primary 4K Blocks	Management Class	Date Last Referenced	Date Migrated	Migration Level	Token for Delete Command
Directory VMSYSU:VMTEST0.							
TEST3	HOLD	176	DEFAULT	11/22/2001	12/01/2001	1	000200000000001DE
TEST5	ML1	1	ML1	12/01/2001	12/01/2001	1	000200000000001DC

Figure 70. Example of an Error Condition Displayed in the File Space Report File

This condition can be caused by the general user erasing a file migrated while DFSMS/VM is down. The delete token is used on the DFSMS DELETE command to remove this data.

- Files that are missing migrated data (see Figure 71 on page 125).

```
The following files are migrated but no migrated data exists for them:
```

Filename	Filetype	Primary 4K Blocks	Management Class	Date Last Referenced
Directory VMSYSU:VMTEST0.				
TEST2	ML2	4	ML2	11/25/2001
TEST3	ML2	1	ML2	11/25/2001
TEST5	ML2	1	ML2	11/25/2001

Figure 71. Example of an Error Condition Displayed in the File Space Report File

This condition can occur if the control file parameters controlling the secondary storage name have been changed without renaming secondary storage. See “Renaming Migration Level 1 and Subdirectory Names” on page 76 for information about renaming of secondary storage. In this case, you update the ML1 directory name. If the error occurred because you were restoring after a DASD failure, then you need to recover those files from a backup.

- Files that are migrated, but data is located in both ML1 and ML2 (see Figure 72 on page 125). This migrated data should be recalled. During recall ML1 is retrieved. If the recalled file does not contain the expected data you must recover the file from backup. This situation is the end result of a storage failure and recovery.

```
The following files are migrated but DFSMS has migrated data in both ML1 and ML2:
```

Filename	Filetype	Primary 4K Blocks	Management Class	Date Last Referenced	Date Migrated	Migration Level	Token for Delete Command
Directory VMSYSU:VMTEST0.FAST1							
ANNABELL	DATA	11	--NULL--	01/06/2001	01/06/2001	1&2;	000200000000001DB
JIM	DATA	21	--NULL--	01/06/2001	01/06/2001	1&2;	000200000000001DA
SAMPLE	EXEC	16	--NULL--	01/06/2001	01/06/2001	1&2;	000200000000001DC

Figure 72. Example of an Error Condition Displayed in the File Space Report File

Example of Reporting on a Storage Group

Figure 73 on page 126 shows an example of the output, returned in a reader file, from the DFSMS REPORT SPACEMANAGEMENT STORGROUP command. The reader file returned upon completion of this command provides you with general information regarding the selected storage group.

```

DFSMS Function Level 221                                01/06/2001  11:36:31
DFSMS REPORT SPACEMANAGEMENT STORGROUP, request identifier 78

DFSMS REPORT SPACEMANAGEMENT STORGROUP processing started for file pool VMSYSU
and storage group 32767

DFSMS REPORT SPACEMANAGEMENT FILESPACE processing started for file pool VMSYSU
and userid VMTEST4

1
Files migrated for this file space:
  Migration Level 1:                                3
  Migration Level 2:                                0
Logical 4K blocks currently in use by this file space: 48
Physical 4K blocks in use by this file space:         48
Physical 4K blocks in use by primary storage for this file space: 45
Physical 4K blocks in use by secondary storage for this file space: 3
  Migration Level 1:                                3
  Migration Level 2:                                0
Physical 4K blocks saved for this file space due to compaction: 0

DFSMS REPORT SPACEMANAGEMENT FILESPACE processing started for file pool VMSYSU
and userid VMTEST5

Files migrated for this file space:
  Migration Level 1:                                8
  Migration Level 2:                                0
Logical 4K blocks currently in use by this file space: 48
Physical 4K blocks in use by this file space:         21
Physical 4K blocks in use by primary storage for this file space: 0
Physical 4K blocks in use by secondary storage for this file space: 21
  Migration Level 1:                                21
  Migration Level 2:                                0
Physical 4K blocks saved for this file space due to compaction: 27

DFSMS REPORT SPACEMANAGEMENT STORGROUP summary for file pool VMSYSU and
storage group 32767

2 Files migrated for this storage group: 11
  Migration Level 1: 11
  Migration Level 2: 0
3 Logical 4K blocks currently in use by this storage group: 96
4 Physical 4K blocks in use by this storage group: 69
5 Physical 4K blocks in use by primary storage for this storage group: 45
6 Physical 4K blocks in use by secondary storage for this storage group: 24
  Migration Level 1: 24
  Migration Level 2: 0
7 Physical 4K blocks saved for this storage group due to compaction: 27

DFSMS REPORT SPACEMANAGEMENT STORGROUP completed with no errors

```

Figure 73. Example of the DFSMS REPORT SPACEMANAGEMENT STORGROUP Command Output

The following is a brief description of the standard messages displayed in the DFSMS REPORT command output (for a detailed description of these messages, see [z/VM: DFSMS/VM Messages and Codes](#)).

- **1** In this section of the report, a file space report appears for each user in the file space.
- **2** FSM3126—Total number of migrated files within this storage group. FSM3163 and FSM3164 display information separating the number of migrated files located in migration level 1 and migration level 2.
- **3** FSM3129—Logical 4K blocks in use by this storage group. Indicating the total number of 4K blocks used by this storage group as if no files migrated.
- **4** FSM3128—Physical 4K blocks in use by this storage group. This number includes the space that migrated files are using in secondary storage in addition to the space that nonmigrated files are using in primary storage.
- **5** FSM3165—Physical 4K blocks in use by primary storage for this storage group. These are files that are not migrated.
- **6** FSM3127—Physical 4K blocks in use by secondary storage for this storage group. This number is the total space used by migrated files for this storage group in secondary storage. Since files are compacted in secondary storage, it should take less room to store the file in secondary storage than it would if this file remained in primary storage. For example, a file that takes up nine 4K blocks in primary storage may

only take up three 4K blocks in secondary storage. This is dependant upon the data itself and the ability of DFSMS/VM to compress it. FSM3163 and FSM3164 display information separating the number of 4K blocks used in migration level 1 and migration level 2.

- **7** FSM3166—Number of physical 4K blocks saved due to file compaction.

Note: One 4K block is the minimum file compaction size; therefore, there is no space savings when single 4K block files are migrated. It is, however, advantageous to migrate single 4K block files when high-speed expensive DASD is used for primary storage and lower cost DASD is used for ML1 and tape for ML2.

The storage group report file also displays the error information provided in the file space report file, shown in [Figure 70 on page 125](#) and [Figure 71 on page 125](#). In addition the storage group report file displays error information shown in [Figure 74 on page 127](#). This information describes migrated files that are present in ML1 or ML2, but for which there are no files in primary storage (see [Figure 74 on page 127](#)). These migrated files are known as *orphans*. These inconsistencies are created by nonstandard operating conditions. For example, a hardware failure results in a recovery process that creates inconsistencies, or DFSMS/VM or the ML2 server is down when files are erased.

The following tokens represent migrated data for which there are no files in primary storage:

Number of 4K Blocks	Date Migrated	Migration Level	Token for Delete Command
115	12/01/2001	1	00020000000001E8
215	02/12/2001	2	00020000000002F7

Figure 74. Example of an Error Condition Displayed in the Storage Group Report File

The DELETE Command

The DELETE command removes:

- ML1 and ML2 entries and data
- ML2 backup entries and data

The type of entries and data deleted by the DFSMS DELETE command is determined by selecting a keyword. The keywords are MIGRATION (which indicates that ML1 and ML2 entries are deleted) and ML2BACKUP (which indicates that ML2 backup entries are deleted). For DFSMS DELETE command syntax and information, see [“DFSMS DELETE” on page 144](#).

Deleting ML1 and ML2 Entries

The MIGRATION keyword provides the capability to delete any inconsistencies between the SFS file pool catalog and ML1 or ML2. These inconsistencies are created by nonstandard operating conditions. For example, a hardware failure results in a recovery process that creates inconsistencies, or DFSMS/VM or the ML2 server is down when files are erased. Eliminate these inconsistencies by performing the following steps:

1. Issue the DFSMS REPORT SPACEMANAGEMENT STORGROUP command. This command's output identifies any inconsistencies with a 16 character token.

Note: If files are migrated and data is contained in both ML1 and ML2, the file must be recalled prior to step 2.

2. Issue the DFSMS DELETE MIGRATION command with the token provided in the DFSMS REPORT command output.

For examples of the error conditions provided in the DFSMS REPORT command output, see [Figure 70 on page 125](#), [Figure 71 on page 125](#), [Figure 72 on page 125](#), and [Figure 74 on page 127](#).

Deleting ML2 Backup Entries

The ML2BACKUP keyword provides storage space savings by deleting obsolete ML2 backup entries. Obsolete ML2 backup entries are present after the SFS FILEPOOL BACKUP command utility tapes are recycled.

Important: Once the backup entries are deleted data **cannot** be recovered, even if backup tape is available.

Part 3. Reference

Chapter 14. DFSMS Commands

This chapter contains reference information about DFSMS commands, ISMF commands, and ISMF line operators. Each command description indicates the command syntax, operands and options; it also lists return codes common to all commands. Usage notes are provided, where applicable.

The command formats are presented in the following order:

- **Command Name:** Identifies the name of the command. The name is also located at the top of the page for easy reference.
- **Function Description:** Describes the general use of the command.
- **Syntax:** Lists the syntax (format) of the command and all possible operands that you can use.
- **Operand and Option Description:** Describes the function of each operand, option, and any values that you can include.
- **Usage Notes:** Contains notes about special uses of the command, its operands, or combinations of commands or operands.
- **Examples:** Provides one or more examples to show how the command is commonly used.

Important

Any userid issuing a DFSMS command or using the ISMF interface (as well as issuing a DFSMSRM command or using the CSL interface), must have an IUCV ANY statement in its directory. See [z/VM: CP Planning and Administration](#) for more information on including this statement in a user directory.

Command Processing Overview

This section briefly describes the output from the different types of DFSMS commands.

Commands Processed in the User Machine

The DFSMS CHECK and COPY commands perform various actions on minidisks. Depending on what options the issuer specifies, the DFSMS COPY or CHECK command output is either stored in a file or displayed on the issuer's console, or both. For example, DFSMS CHECK 191 (FILE displays the results on the console and puts them in a file.

Commands that Produce Reports

ALTER, CONVERT, DELETE, MANAGE, MIGRATE, MOVE, RECALL, and REPORT are commands that perform actions on minidisks or SFS files (either in primary or secondary storage) or report on the status of them. The WAIT or NOWAIT option can be used with the ALTER, MANAGE, MIGRATE, MOVE, and RECALL commands. If the WAIT option has not been specified, a unique request identifier is displayed on the issuer's console when the command has been accepted by DFSMS/VM for processing. The request identifier is also logged with many of the messages written to the log file associated with the execution of the command. When command execution completes, a report file which contains messages associated with the execution of the command is sent to command issuer's reader.

If the WAIT option has been specified, the actual command execution is similar to the other commands. The difference is that the command issuer does not receive control of the z/VM session until command processing is complete and the report file is sent to the command issuer's reader. The report file contains detailed information about the execution of the command and should be reviewed. When the command issuer receives control, a message (which includes the request identifier) is displayed on their console indicating that the request is complete.

For functional and reporting commands, files that are sent to the command issuer use the following naming convention:

- The file name is the APPC/VM global resource ID that is specified in the DFSMS control file.
- The file type begins with T followed by the request ID.

Operational Commands

These commands change or query the operation of the DFSMS/VM virtual machines. DFSMS ACTIVATE, DISCARD, QUERY, and STOP are operational commands.

The output from an operational command is displayed on the console of the command issuer. No request identifiers are displayed for these commands.

Return Codes

Table 6 on page 132 lists the return codes for DFSMS commands.

<i>Table 6. DFSMS Command Processor Return Codes</i>	
Return Code	Meaning
0	Successful processing.
4	SKIP option has been ignored during DFSMS COPY. A problem has been detected with the minidisk during DFSMS CHECK.
24	Syntax error occurred.
28	File or directory not found or already exists, or insufficient authority.
32	File not in expected format, or file does not contain the expected information, or an attempt was made to execute a LOADMOD command while in CMS subset mode.
36	File mode not accessed. Problem with source or target minidisk during DFSMS COPY.
40	No default file pool assigned.
41	Insufficient storage.
55	Communications error.
70	File sharing conflict. This includes locking conflicts and failures caused by uncommitted changes.
74	Requested function not available for minidisks.
76	Authorization error.
88	Block size invalid for device during DFSMS COPY.
99	Required system resource not available (the CSL library is not installed, or server is not available).
100	I/O error has occurred during DFSMS COPY. Device error has occurred during DFSMS CHECK.
104	Functional error for which the system is responsible. Insufficient storage for DFSMS COPY or DFSMS CHECK.
200	Allocation error has occurred for DFSMS CHECK.
201	Allocation error has occurred for DFSMS CHECK.
256	Unexpected error for which the system is responsible. The output minidisk not large enough for DFSMS COPY.
300	Internal logic error has occurred during DFSMS COPY.
301	Internal logic error has occurred during DFSMS COPY.

Table 6. DFSMS Command Processor Return Codes (continued)	
Return Code	Meaning
302	Internal logic error has occurred during DFSMS COPY.
303	Internal logic error has occurred during DFSMS COPY.
304	Internal logic error has occurred during DFSMS COPY.

Online HELP for Messages

The z/VM HELP facility conveniently displays information about DFSMS/VM messages. Help text for messages is displayed when you invoke:

```
HELP cccnnns
```

or

```
HELP cccmmnnns
```

Where:

ccc

is the component ID of the message (FSM, IGD, DGT).

mmm

is the optional module identifier.

nnnn

is the message number.

s

is the message severity (I, W, E).

See the [z/VM: CMS Commands and Utilities Reference](#) for additional information about the z/VM HELP facility.

Online HELP for DFSMS Commands

The z/VM HELP facility conveniently displays information about DFSMS commands.

The following forms of help are available in DFSMS/VM Function Level 221:

- A menu panel of DFSMS commands when you invoke:

```
help dfsms
```

- A command help panel when you invoke:

```
help dfsms command
```

or

```
help dfsmsim command
```

where *command* is a DFSMS command. For example, entering:

```
help dfsms activate
```

results in a help panel describing the DFSMS ACTIVATE command. See [Figure 75 on page 134](#).

```

DFSMS ACTIVATE      ALL Help Information      line 1 of 44
(c) Copyright IBM Corporation 1993
(adapted from IBM Form SH35-0111)

DFSMS ACTIVATE

The DFSMS ACTIVATE command provides you with the ability to create an
active configuration from a source configuration.

Note: You can also activate a source configuration through the
activate function within the ISMF Configuration Application.

The command syntax is:

+--- DFSMS ACTIVATE Command -----+
|                                   |
| >>--DFSMS ACTivate--fn--ft--dirid----->< |
|                                   |
+-----+

Operands

fn ft
are the standard file name and file type for CMS files, each con-
sisting of 1-8 characters. This fully qualified file name and
file type identifies the source configuration file to be acti-
vated.

```

Figure 75. Example DFSMS/VM HELP Panel

See the [z/VM: CMS Commands and Utilities Reference](#) for additional information about the z/VM HELP facility.

Pattern Matching

Two special characters, referred to as pattern-matching characters, can be used in the file name (*fn*) and file type (*ft*). The power of pattern-matching characters is that they make one file name or file type match a number of files. Pattern-matching characters are used with the DFSMS ALTER, MIGRATE and RECALL commands to specify a subset of your files, rather than just one file.

The two characters are:

%

Matches any one character in the file name or file type

Matches any number of characters in the file name or file type

Here is an example. You have files named:

```

NETLOG AFILE A1
MYLOG AFILE A1
TLOG AFILE A1
NAMELOG3 AFILE A1

```

A file name specified as `"*LOG"` matches all four files. A file name specified as `"%LOG"` matches only `"TLOG"`.

Using DFSMS Commands

DFSMS commands assist you when performing minidisk management and space management tasks. You can enter the commands from either the command line in ISMF, when preceded by CMS, or in the CMS environment.

The following notation is used in the command syntax of this book:

- Truncation

Where truncation of a command name is permitted, the shortest acceptable version of the command is represented by uppercase letters. (Remember, however, that DFSMS commands can be entered with any combination of uppercase and lowercase letters.)

For example:

DFSMS CONvert

This means that the command can be entered as: DFSMS CONVERT, dfsms CON, dfsms CONv, dfsms CONVe, and so on.

Operands and options are specified in the same manner. If no minimum truncation is permitted, the entire word, as represented by all upper case letters, must be entered.

- An option that is the default is indicated by an underscore. If you desire the underscored choice, you need not specify the option when the command is entered.

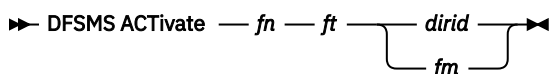
DFSMS ACTIVATE

The DFSMS ACTIVATE command provides you with the ability to create an active configuration from a source configuration. The active configuration provides the storage management policies to DFSMS/VM.

Note: You can also activate a source configuration through the activate function within the ISMF Configuration Application. For additional information about activating configurations, see [Chapter 5, “Configurations,”](#) on page 53.

The command syntax is:

DFSMS ACTIVATE Command



Operands

fn ft

are the standard file name and file type for CMS files, each consisting of 1–8 characters. This file name and file type identifies the source configuration file to be activated.

dirid

is the name of the SFS directory in which the file to be processed is located. If the directory is a directory control directory, write authority is required to activate.

fm

indicates a letter between **A** and **Z** that specifies an accessed SFS directory. The *+fm.ni[.ni+1... ni+7]* forms or the *-fm[.ni.ni+1... ni+7]* forms can also be used. For additional information about SFS naming conventions, refer to the *z/VM: CMS Commands and Utilities Reference*. The file mode number is optional. A file mode of asterisk (*) is not accepted.

Example

If you want to have a source configuration (for example, *CONFIG NEW1* located in the SFS directory *VMSYSU:MELANIE.SRCCNFG*) to be the active configuration, enter:

```
DFSMS ACTIVATE CONFIG NEW1 VMSYSU:MELANIE.SRCCNFG
```

Figure 76 on page 135 shows a sample of the output from the DFSMS ACTIVATE command when there is a REXX exit, but no module exit.

```
FSMUAT8070I File CONFIG NEW1 VMSYU:MELANIE.SRCCNFG activated
FSMUAT8072I ACS module exit not activated
FSMUAT8071I ACS REXX exit activated
```

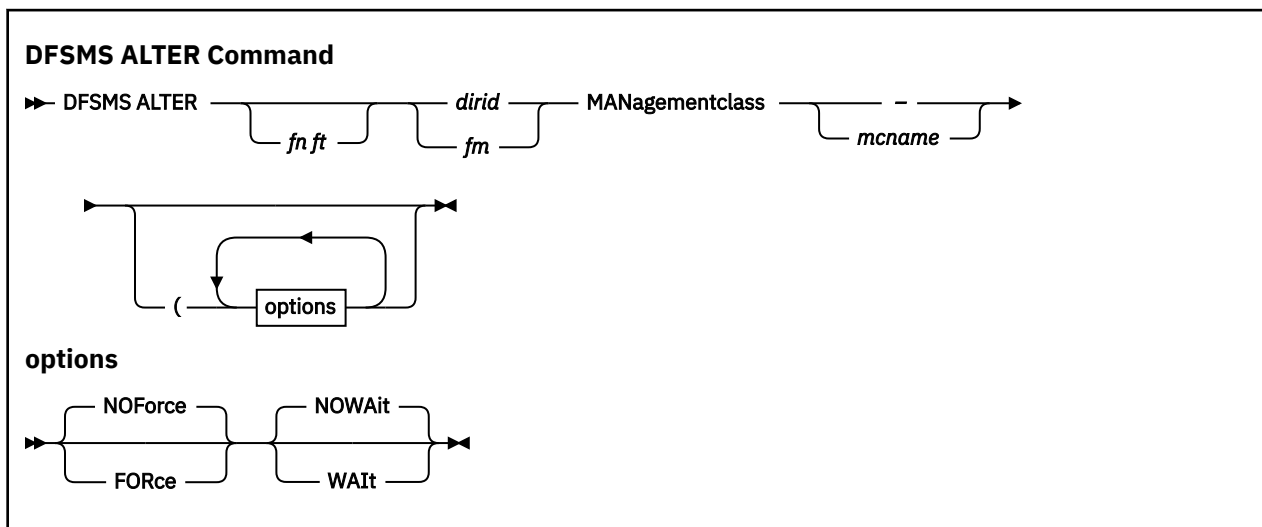
Figure 76. Example of the DFSMS ACTIVATE Command Output

The messages for successful activation of a configuration can vary depending on whether or not the installation is using the ACS REXX or ACS module exits.

DFSMS ALTER

The DFSMS ALTER command provides you with the ability to change management classes of SFS files and directories, including top-level directories. Because you can specify a pattern-matching character in the file ID, this command has the capability of altering the management class of multiple files. Only base files, however, are matched during pattern-matching processing for the DFSMS ALTER command.

The command syntax is:



Operands

fn ft

are the standard file name and file type for CMS files, each consisting of 1–8 characters. This file name and file type identify the file that is assigned a new management class name. When the file name and file type are not specified, it is assumed that DFSMS ALTER command processing assigns a management class name to the directory. Pattern-matching characters are allowed in the file name and file type fields and are used only for base files. Pattern-matching characters are not used for directories, external objects, and aliases. The DFSMS ALTER *dirid* form of the command must be used to change the management class of a directory.

dirid

is the name of the SFS directory in which the file to be processed is located.

fm

indicates a letter between **A** and **Z** that specifies an accessed SFS directory. The *+fm.ni[.ni+1... ni+7]* forms or the *-fm[.ni.ni+1... ni+7]* forms can also be used. For additional information about SFS naming conventions, refer to the [z/VM: CMS Commands and Utilities Reference](#). The file mode number is optional. A file mode of asterisk (*) is not accepted.

MANagementclass

is a required positional keyword.

mcname

is the name of the management class assigned to the files or directory. This name must comply with management class naming conventions (including when the FORCE option is issued). This is a required parameter. If "-" is specified for this field, the *NULL management class* is assigned to the files or directory.

Options

NOForce

is an optional parameter indicating that the management class assignment should *not* take place if the specified management class does not exist in the active configuration. If a management class of "-" is specified, this keyword is ignored. The NOFORCE parameter is the default and cannot be used with the FORCE parameter.

FORce

is an optional parameter indicating that the management class assignment should take place even if the specified management class does not exist in the active configuration. If a management class of "-" is specified, this keyword is ignored. The FORCE parameter cannot be used with the NOFORCE parameter.

Important: Any file assigned a management class not in the active configuration cannot be managed by DFSMS/VM; however, if a configuration is activated containing this management class, the files can then be managed by DFSMS/VM.

NOWait

is an optional parameter indicating that the command issuer does not want to wait until completion of DFSMS ALTER command processing. Control is returned to the command issuer when the command is received by DFSMS/VM. The NOWAIT parameter is the default and cannot be used with the WAIT parameter.

WAIt

is an optional parameter indicating that the command issuer wants to wait until completion of DFSMS ALTER command processing. Control is not returned to the command issuer until command processing is complete. The WAIT parameter cannot be used with the NOWAIT parameter.

Usage Notes

Upon successful completion of DFSMS ALTER command processing, a reader file is returned indicating the old and new management class of any altered files or directory.

Example

If you want to change the management class of all files with the file type of *REPORT* located in the SFS directory *VMSYSU:DEPT.STATUS* to the *MIGNOEXP* management class and you want to wait until command processing is complete, enter:

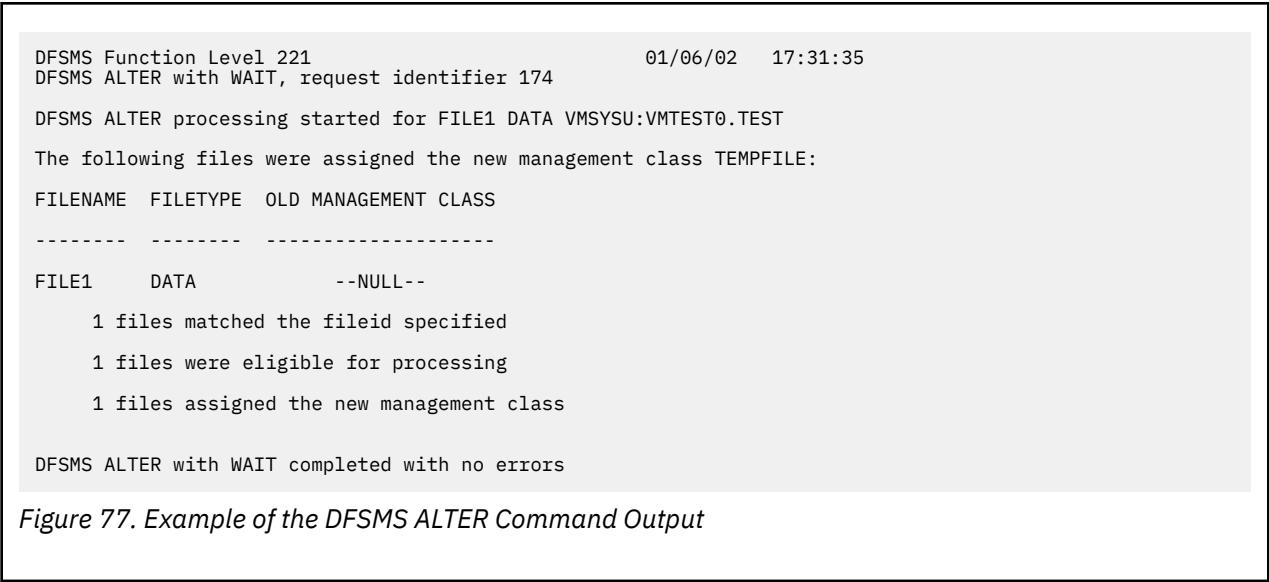
```
DFSMS ALTER * REPORT VMSYSU:DEPT.STATUS MAN MIGNOEXP (WAI
```

If you want to change the management class of the *STATUS* directory within the *VMSYSU* file pool (that is, *VMSYSU:DEPT.STATUS*) to the *MIGNOEXP* management class, and you did not want to wait until command processing is complete, enter:

```
DFSMS ALTER VMSYSU:DEPT.STATUS MAN MIGNOEXP
```

[Figure 77 on page 138](#) shows a sample of the output from the DFSMS ALTER command.

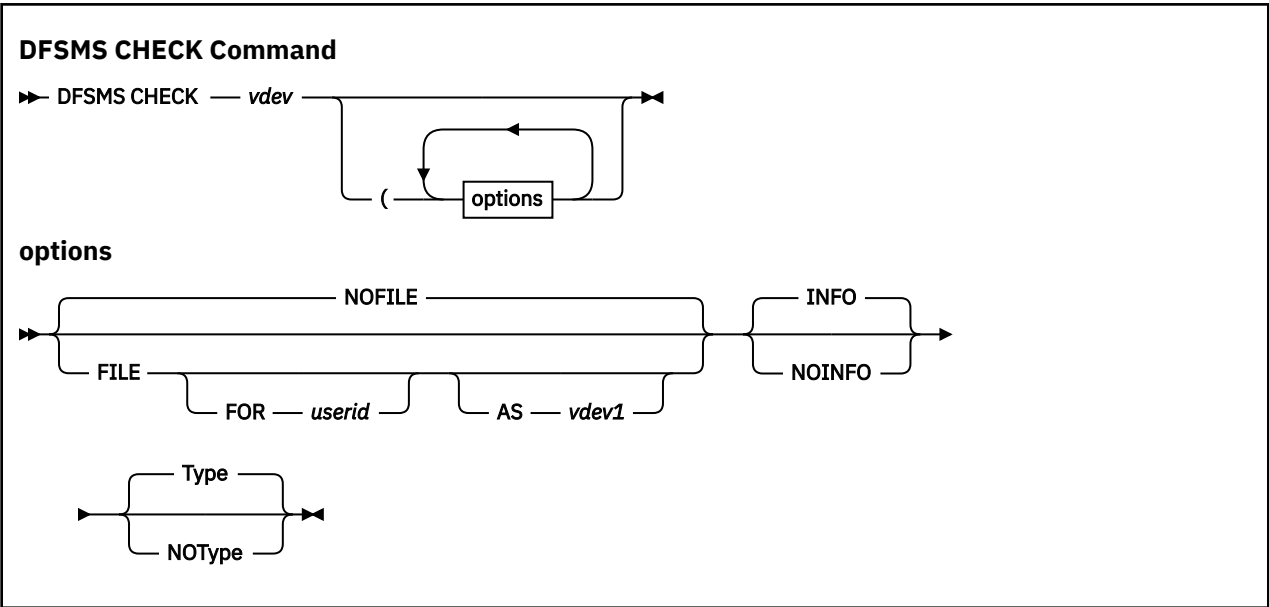
CHECK



DFSMS CHECK

The DFSMS CHECK command verifies the integrity of a minidisk. This command executes in the user virtual machine and supports only CMS-formatted minidisks.

The command syntax is:



Operands

vdev
is the minidisk virtual address you want to check. The address consists of 1–4 hex digits.

Options

NOFILE
specifies that messages are not written to a file. The NOFILE parameter is the default and cannot be used with the FILE parameter.

FILE

allows messages to be placed in a file for you. The file ID for the message file is *userid vdevCHCK A*, where *userid* is your VM user ID and *vdev* is the virtual address of the target minidisk you are checking (unless either or both of the FOR and AS options are specified). The FILE parameter cannot be used with the NOFILE parameter.

FOR *userid*

allows you to specify a file name other than the default file name used by the FILE option. *userid* is any user ID you specify.

AS *vdev1*

allows you to specify a file type other than the default file type used by the FILE option. *vdev1* is a minidisk virtual address you specify and must be 1–4 hex digits. The file ID for the file containing DFSMS CHECK messages is *userid vdev1CHCK A*, unless the FOR option has been specified to select a different user ID.

INFO

specifies that informational messages for the CHECK command are generated. The INFO parameter is the default and cannot be used with the NOINFO parameter.

NOINFO

prevents the display of informational messages for the DFSMS CHECK command. The NOINFO parameter cannot be used with the INFO parameter.

Type

specifies that messages are displayed on the console. The TYPE parameter is the default and cannot be used with the NOTYPE parameter.

NOTYPE

prevents the display of messages on the console. The NOTYPE parameter cannot be used with the TYPE parameter.

Usage Notes

- The DFSMS CHECK command can be invoked from either the CMS environment or an ISMF panel.
- If file output is requested, the issuer's A-disk or directory must be linked in read/write mode.

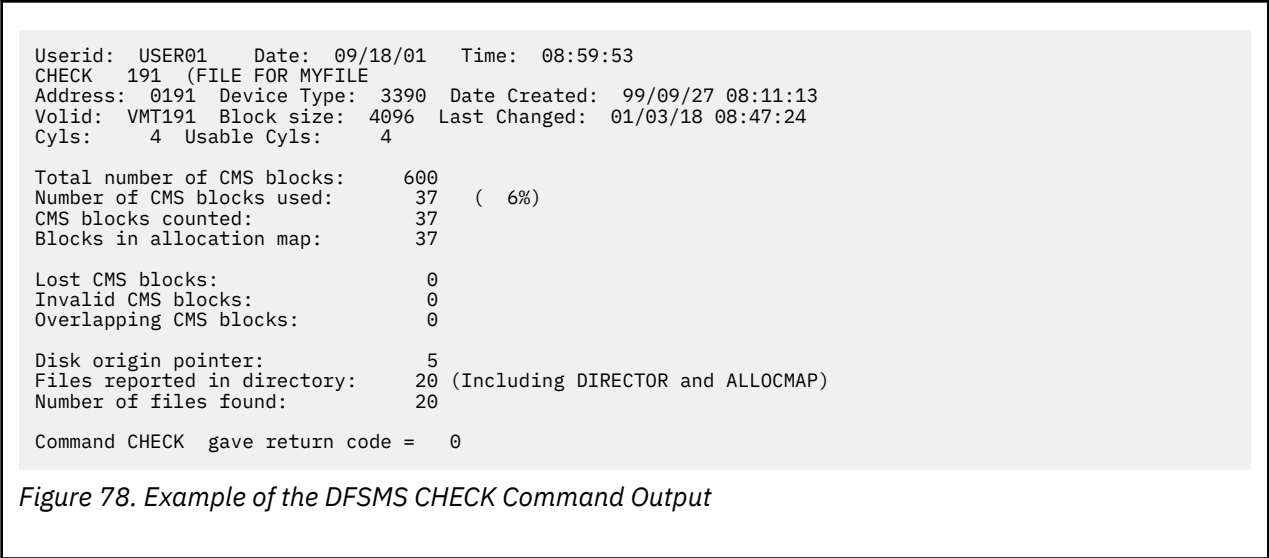
Example

To verify the integrity of your 191 minidisk and place the messages in a file named MYFILE 0191CHCK A. Enter:

```
DFSMS CHECK 0191 (FILE FOR MYFILE
```

[Figure 78 on page 140](#) shows a sample of the output from the DFSMS CHECK command.

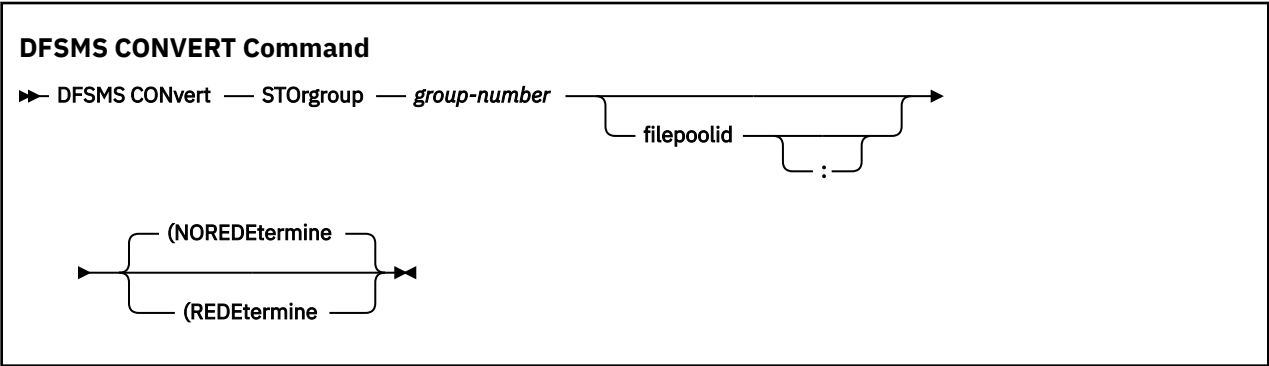
CONVERT



DFSMS CONVERT

The DFSMS CONVERT command assigns management classes to SFS files and directories that are without management classes and optionally reassigns new management classes to files according to ACS processing.

The command syntax is:



Operands

- group-number**
specifies the storage group containing the files and directories against which ACS processing is invoked. Any defined storage group except storage group 1 can be specified in this command.
- filepoolid**
specifies the file pool in which the storage group resides. If not specified, the current file pool ID of the command issuer is used.

Options

- NOREDEtermine**
indicates that only files and directories that have no management classes are converted. The NOREDETERMINE parameter is the default and cannot be used with the REDETERMINE parameter.
- REDEtermine**
specifies that all files and directories in the storage group are converted even if they already have management classes. The REDETERMINE parameter cannot be used with the NOREDETERMINE parameter.

Usage Notes

- The DFSMS CONVERT command can require significant processing time; therefore, you should run this command during off-shift hours.
- The DFSMS CONVERT command can be cancelled by issuing either the DFSMS DISCARD command, the ISMF DISCARD command from an ISMF panel, or the DFSMS STOP SMS (IMMEDIATE command. The request ID of the DFSMS CONVERT command is required when using the DISCARD command. The request ID is returned to the issuer of the CONVERT command when the command is accepted by DFSMS/VM. You can also acquire the request ID via the ISMF QUERY command or DFSMS QUERY command.
- If two DFSMS CONVERT commands are executed concurrently against the same storage group, the second DFSMS CONVERT command fails. Also, if one DFSMS CONVERT and one DFSMS MANAGE command are executed concurrently against the same storage group, the second command fails. The first command executed in both instances continues processing.
- If you make changes to ACS processing, specify the REDETERMINE option to reassign management classes based on the new ACS process. All files and directories in a storage group are updated.

Example

You activated a new source configuration containing a new management class and updated your ACS processing to assign this new management class. You also need to run ACS processing against all files and directories in storage group 3 to reflect this new configuration. Enter:

```
DFSMS CONVERT STORGROUP 3 (REDETERMINE
```

Figure 79 on page 141 shows a sample of the output from the DFSMS CONVERT command. However, this sample does not contain all possible message combinations.

```
DFSMS Function Level 221                      01/18/02  09:22:38
DFSMS CONVERT, request identifier 273

DFSMS CONVERT processing started for storage group 3 in file pool VMSYSU

FSMDAM3230I Directory VMSYSU:MAINT. assigned management class ACCNTDPT
FSMDAM3171I File RECORDS ISPPROF VMSYSU:MAINT. assigned management class ACCNTDPT
FSMDAM3171I File EMPLOYEE ISPPROF VMSYSU:MAINT. assigned management class ACCNTDPT
FSMDAM3171I File DGTTABL MACLIB VMSYSU:MAINT. assigned management class ACCNTDPT
FSMDAM3171I File DGT0PROF ISPPROF VMSYSU:MAINT. assigned management class ACCNTDPT
FSMDAM3230I Directory VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMA11 $PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMA11 PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMB11 $PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMB11 PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMD11 $PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMD11 PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMF11 $PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMF11 PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMH11 $PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMH11 PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMK11 $PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVMK11 PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT
FSMDAM3171I File 5VMVML11 $PPF VMSYSU:MAINT.SYSTEM. assigned management class ACCNTDPT

      16 files and directories converted
      0 files and directories could not be converted

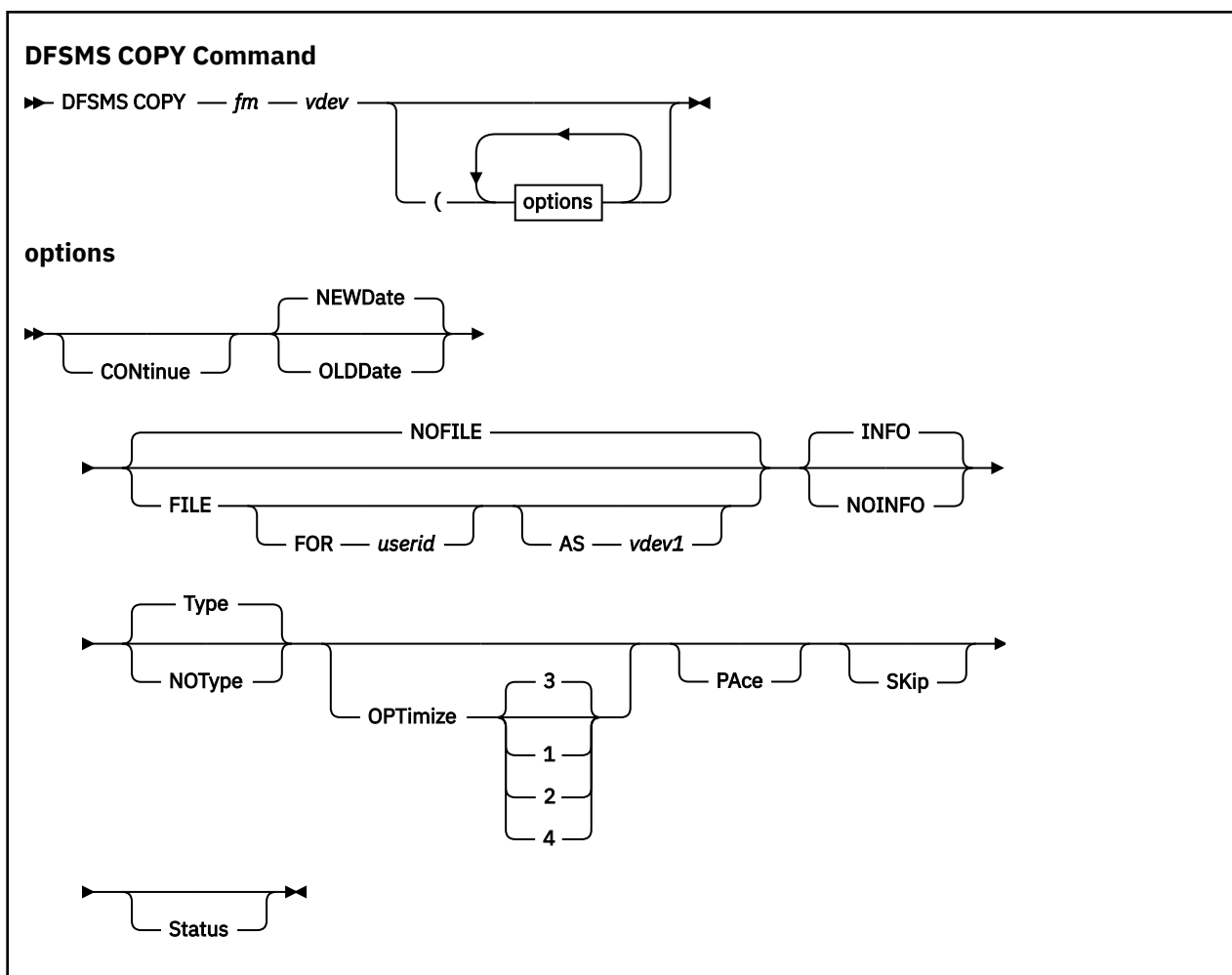
DFSMS CONVERT completed with no errors
```

Figure 79. Example of the DFSMS CONVERT Command Output

DFSMS COPY

The DFSMS COPY command copies a minidisk to an unformatted target minidisk. You specify the source minidisk file mode and the target virtual address as part of the command. DFSMS COPY does not require that the target minidisk be formatted prior to a copy operation because it formats the minidisk and copies data to it within the same operation. However, once the copy operation is completed by DFSMS COPY, you must run CMS FORMAT with the RECOMP option against the target minidisk to make sure that the allocation map is updated. You also need to make CP directory updates. This command executes in the user's virtual machine.

The command syntax is:



Operands

fm

is the file mode associated with the source minidisk.

vdev

is the virtual address of the target minidisk and must consist of 1–4 hex digits.

Options

CONTINUE

allows copying to continue even if permanent I/O errors occur during copying. The default is to stop the copy operation on any permanent I/O error.

NEWDate

specifies that today's date is used as the creation date in the label of the target minidisk. The NEWDATE parameter is the default and cannot be used with the OLDDATE parameter.

OLDDate

specifies that the creation date from the source minidisk is the creation date in the label of the target minidisk. The OLDDATE parameter cannot be used with the NEWDATE parameter.

NOFILE

specifies that messages from the DFSMS COPY command are not written to a file. The NOFILE parameter is the default and cannot be used with the FILE parameter.

FILE

specifies that messages from the DFSMS COPY command are placed in a file. The file ID for the message file is *userid vdevCOPY A*, where *userid* is your VM user ID and *vdev* is the virtual address of the target minidisk. If you want to use another file ID for the message file, you can use the FOR *userid* and AS *vdev1* options. The FILE parameter cannot be used with the NOFILE parameter.

FOR *userid*

specifies a file name other than the default file name used by the FILE option. *userid* is any user ID that you specify.

AS *vdev1*

specifies a file type other than the default file type used by the FILE option. *vdev1* is a minidisk virtual address that you specify, and it must be 1–4 hex digits. The file ID for the file containing DFSMS COPY messages is *userid vdev1COPY A*, unless the FOR option has been used to select a different user ID.

INFO

specifies that informational messages for the DFSMS COPY command are generated. The INFO parameter is the default and cannot be used with the NOINFO parameter.

NOINFO

prevents the display of informational messages for the DFSMS COPY command. The NOINFO parameter cannot be used with the INFO parameter.

Type

specifies that messages produced by the DFSMS COPY command are displayed on the console. The TYPE parameter is the default and cannot be used with the NOTYPE parameter.

NOType

prevents the display of messages produced by the DFSMS COPY command to the console. The NOTYPE parameter cannot be used with the TYPE parameter.

OPTimize

specifies the unit size of data transfer:

1

transfers data in single track units for count key data (CKD) devices.

2

transfers data in two track units for CKD devices.

3

transfers data in five track units for CKD devices. This is the default.

4

transfers data in one cylinder units for CKD devices.

Note: A maximum of 255 blocks are transferred during one I/O operation. Therefore, if the requested OPTIMIZE level contains greater than 255 blocks, multiple I/O operations are required. Use options 1, 2, or 3 to move block sizes of 2KB or smaller on either 3380 or 3390 devices.

PAce

causes a two-second delay between each data unit transfer. Use this option to minimize the impact on other users. The default is no delay.

SKip

checks within units of data transfer for any blocks in use by the CMS file system. If any blocks are in use, read I/O is not performed. You must have read/write access to the source minidisk to use this option.

SStatus

allows the display of periodic progress messages at the console as the data is copied. The default does not issue progress messages to the console.

Restrictions

DFSMS COPY has the following restrictions:

- The target minidisk must hold at least as many CMS blocks as the source minidisk contains.

DELETE

- On a minidisk that has been processed by CMS FORMAT with the RECOMP option, data outside of the CMS-formatted portion of the minidisk cannot be copied to the target minidisk.
- The block size of the source minidisk is preserved on the target minidisk. Reblocking is not supported.
- If you specify the SKIP option, you must have read/write access to the source minidisk.
- The CMS block size of the source minidisk must be supported on the target device type.
- CMS must have enough virtual storage under 16 MB available to access the source and target disks.

Note: Because the CMS file system requires file status and control information to reside below 16 MB in virtual storage, there is a practical limitation on the size of CMS minidisks. As a minidisk increases in size, or more files reside on the disk, the amount of virtual storage associated with the disk for CMS file system status and control increases in storage below 16 MB. The current ECKD DASD limitation is 32767 cylinders for a 3390 disk on an IBM TotalStorage DASD subsystem, or about 22 GB of data. IBM suggests that customers defining FBA SCSI disks for use by CMS should set a practical limit of about 22 GB. If larger disks are defined, they should be limited to contain very few files, or the CMS file system may not be able to obtain enough virtual storage below 16 MB to format or access those disks. For more information, see the ACCESS command in the [z/VM: CMS Commands and Utilities Reference](#). The maximum size for FBA SCSI disks supported for use by CMS or GCS is 381 GB.

Usage Notes

- The DFSMS COPY command offers a subset of the function provided by DFSMS MOVE and ISMF MOVE commands. Besides freeing your virtual machine during the copy operation, DFSMS MOVE and ISMF MOVE perform CP directory updates and adjust the sizes of the allocation maps on the target disks automatically.

Example

To copy a minidisk accessed as a file mode to the minidisk linked as virtual address 192, enter:

```
DFSMS COPY B 0192 (FILE
```

Figure 80 on page 144 shows a sample of the output from the DFSMS COPY command.

```
02/18/02 09:45:06      Userid: VMTEST0   Date: 02/18/02   Time: 09:45:06
02/18/02 09:45:06      COPY      B      192 (
02/18/02 09:45:18      Command COPY      gave return code = 0
```

Figure 80. Example of the DFSMS COPY Command Output

DFSMS DELETE

The DFSMS DELETE command removes data objects from DFSMS/VM-owned storage. A data object is defined as one of the following:

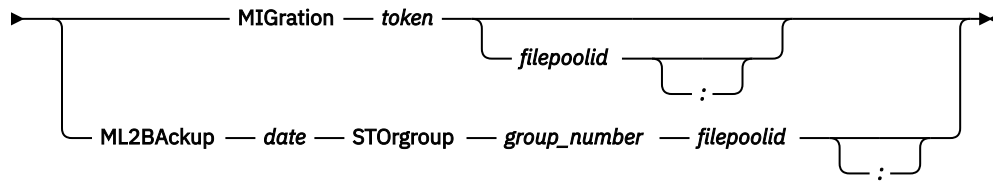
- Migration level 1 (ML1) migration entries and data
- Migration level 2 (ML2) migration entries and data
- ML2 backup entries and data

When issuing the DFSMS DELETE command, you have two keywords to choose from: the MIGRATION keyword and the ML2BACKUP keyword. The MIGRATION keyword is used in conjunction with the DFSMS REPORT command. A portion of the output generated by the DFSMS REPORT command lists, when applicable, any inconsistencies between the SFS file pool catalog and DFSMS secondary storage. The DFSMS DELETE MIGRATION command resolves these inconsistencies. The ML2BACKUP keyword provides the capability to delete obsolete ML2 backup entries.

The command syntax is:

DFSMS DELETE Command

➡ DFSMS DElete ➡

**Operands for the MIGRATION Keyword*****token***

is the 16-character token of the file to be deleted, as identified in the output of the DFSMS REPORT command. DFSMS/VM verifies that the token really identifies an out-of-sync condition before proceeding with DFSMS DELETE command processing. An out-of-sync condition, for example, is when a migrated file exists, but a corresponding primary file does not exist.

filepoolid

identifies the file pool where the file was migrated from. If the file pool is not specified, the user's current file pool ID is used. An optional colon is allowed after the file pool ID.

Operands for the ML2BACKUP Keyword***date***

is a date associated with file backups created by the SFS FILEPOOL BACKUP command. When the SFS FILEPOOL BACKUP command utility tapes are recycled, the DELETE command is used to erase all DFSMS ML2 backup entries for the given file pool and storage group prior to the date specified by this field. This is an eight character field of the form *yyyymmdd*, where *yyyy* is the year, *mm* is the month, and *dd* is the day (for example, November 15, 1992 is specified *19921115*).

group_number

is the storage group for which DELETE processing of SFS backup utility entries takes place. Any defined storage group except storage group 1 can be specified in this command. If storage group 1 or an undefined storage group is specified, an error results and DFSMS DELETE command processing terminates.

filepoolid

identifies the file pool that was backed up from which the ML2 backup entries are to be deleted. This parameter is required, and an optional colon is allowed after the file pool ID.

Usage Notes

- When using the DFSMS DELETE command to resolve out-of-sync conditions where both ML1 and ML2 exist, the ML2 version is deleted first. Before performing this deletion, recall and verify that the ML1 version of this file is what you want.
- When SFS storage group backup completes, a message is issued containing the time and date of the completed backup. It is recommended that you retain this information for later use when deleting obsolete ML2 backup entries from the ML2 inventory.
- As SFS backup tapes are recycled, the DFSMS DELETE ML2BACKUP command is used to remove obsolete entries from the ML2 inventory and database. If the DFSMS DELETE ML2BACKUP command is not used, space in the ML2 repository cannot be reclaimed.

Note: When a backup entry is present for a file, the file is not erased from ML2 even if the file has been recalled, until the DFSMS DELETE ML2BACKUP command is used.

- The ML2 backup entries can *only* be deleted via the DFSMS DELETE ML2BACKUP command.

DISCARD

- The DFSMS DELETE ML2BACKUP command eliminates all ML2 backup entries prior to the specified date. Be careful when executing this command, especially when certain backups need to be retained for extended periods of time. For example, you cannot delete the weekly backup while saving the year-end backup. When DFSMS DELETE ML2BACKUP is completed, there are no backup references to files in ML2. They are erased from ML2 thus reclaiming space.

Example

To delete a file, from secondary storage, with the token ID of 00020000000001EB, enter:

```
DFSMS DELETE MIG 00020000000001EB
```

To delete all ML2 backup entries in the MFG file pool within storage group 4, created before October 31,2001, enter :

```
DFSMS DELETE ML2BA 20011031 STO 4 MFG
```

Figure 81 on page 146 shows an example of the output from the DFSMS DELETE MIGRATION command.

```
DFSMS Function Level 221                      01/03/02   11:09:15
DFSMS DELETE MIGRATION, request identifier 6

FSMDEL3172I File with token 0002000000000202 deleted from ML1
DFSMS DELETE MIGRATION completed with no errors
```

Figure 81. Example of the DFSMS DELETE MIGRATION Command Output

Figure 82 on page 146 shows an example of the output from the DFSMS DELETE ML2BACKUP command.

```
DFSMS Function Level 221                      01/01/02   10:41:48
DFSMS DELETE ML2BACKUP, request identifier 6

DFSMS DELETE ML2BACKUP processing started for storage group 4 in file pool
MFG for entries with dates previous to 20011031

FSMDTC3115I 1033 backup entries were deleted from ML2 storage
DFSMS DELETE ML2BACKUP completed with no errors
```

Figure 82. Example of the DFSMS DELETE ML2BACKUP Command Output

DFSMS DISCARD

The DFSMS DISCARD command provides you with the ability to discard requests. If you do not have the request ID, it can be obtained with the DFSMS QUERY REQUESTs command. The requests that can be discarded are:

- ALTER
- CHECK (minidisk)
- CONVERT
- DELETE
- MANAGE
- MIGRATE
- MOVE (minidisk)
- RECALL (command recall only)
- REPORT

Messages are issued directly to the command issuer's console.

The command syntax is:

DFSMS DISCARD Command

➤ DFSMS DISCard — *request_id* ➤

Operands***request_id***

is the request identifier of command you want to discard. The request identifier can be identified via the DFSMS QUERY REQUEST command.

Usage Notes

The DFSMS DISCARD command issuer does not regain control until command processing is complete.

The DFSMS DISCARD command discards any items that have not started processing; however, only the DFSMS MANAGE and DFSMS CONVERT command can be discarded during processing. Other than DFSMS MANAGE and CONVERT processing, the DFSMS DISCARD command can only discard items that have not yet started processing. For example, if DFSMS MIGRATE * * applies to 20 files and 5 have started processing, 15 can be discarded, but the other 5 complete processing.

Example

A DFSMS MANAGE command is issued, the request identifier given is 24. If you wanted to discard the manage request you would enter:

```
DFSMS DISCARD 24
```

Figure 83 on page 147 shows an example of the output from the DFSMS DISCARD command.

```
FSMDIS3265I Request identifier 24 was discarded successfully
```

Figure 83. Example of the DFSMS DISCARD Command Output

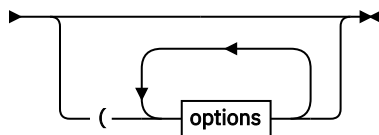
DFSMS MANAGE

The DFSMS MANAGE command causes DFSMS/VM to erase expired files and migrate or move files in a SFS storage group according to the attributes of each file's management class. Files are *migrated* from primary storage to either migration level 1 (ML1) or migration level 2 (ML2) and files are *moved* from ML1 to ML2. For additional information, see [Chapter 13, "Managing Storage Using DFSMS/VM,"](#) on page 117.

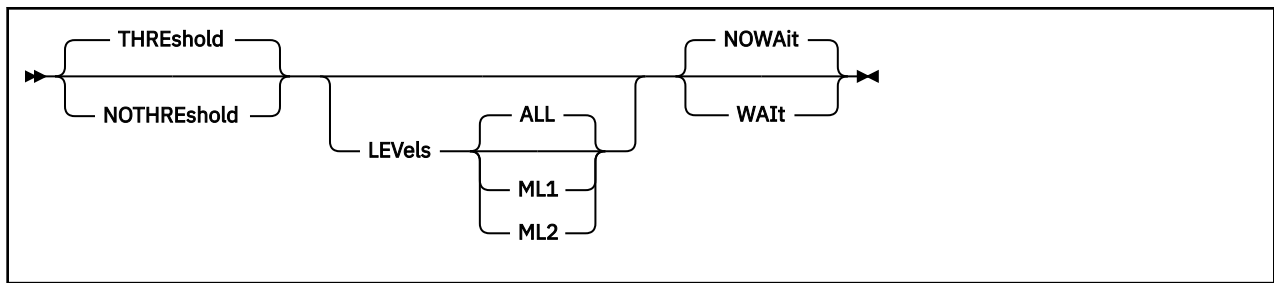
The command syntax is:

DFSMS MANAGE Command

➤ DFSMS MANAge STOrgroup — *group_number* — *filepoolid* —



options



Operands

group-number

specifies the storage group number containing the files and directories to be processed. Any defined storage group except storage group 1 can be specified in this command. If storage group 1 or an undefined storage group is specified, an error results and DFSMS MANAGE command processing terminates.

filepoolid

specifies the file pool in which the storage group resides. If not specified, the current file pool ID of the command issuer is used. An optional colon is allowed after the file pool ID.

Options

THREshold

indicates that DFSMS/VM should use both the high and low threshold keywords that are specified in the DFSMS/VM control file. The values determine the percentage of space you want to manage. The THRESHOLD parameter is the default and cannot be used with the NOTHRESHOLD parameter.

NOTHREshold

indicates that DFSMS/VM should ignore the high and low thresholds that are specified in the DFSMS/VM control file. All expired files are erased, and all files eligible for migration are migrated. NOTHRESHOLD cannot be used with the THRESHOLD parameter.

LEVels

indicates whether migration processing is allowed for ML1, ML2, or both ML1 and ML2. This parameter controls movement to particular migration levels, thereby allowing an installation to control tape mount activity. The options for this parameter are:

ALL

migration is allowed to all migration levels (that is, migration from primary storage to ML1, migration from primary storage to ML2, and a move from ML1 to ML2). Expiration is performed for all eligible files in primary, ML1, and ML2 storage. This is the default. If ML2 is not defined for the installation, eligible files are migrated to ML1.

ML1

migration is allowed only from primary storage to ML1. No moves are allowed from ML1 storage to ML2. Expiration is performed for all eligible files in primary, ML1, and ML2 storage.

ML2

migration is allowed only from primary to ML2 and moves are allowed from ML1 to ML2. No migration is allowed from primary storage to ML1. Expiration is performed for all eligible files in primary, ML1, and ML2 storage. If ML2 is not defined for the installation, command processing stops.

NOWAit

is an optional parameter indicating that the command issuer does not want to wait until completion of DFSMS MANAGE command processing. Control is returned to the command issuer when the command is received by DFSMS/VM. The NOWAIT parameter is the default and cannot be used with the WAIT parameter.

WAIT

is an optional parameter indicating that the command issuer wants to wait until completion of DFSMS MANAGE command processing. Control is not returned to the command issuer until command processing is complete. The WAIT parameter cannot be used with the NOWAIT parameter.

Usage Notes

- If during DFSMS MANAGE command processing DFSMS/VM finds a file or directory that does not have a management class, ACS processing is invoked. ACS processing assigns a management class to the file or directory, and then the DFSMS MANAGE command completes processing of this file.
- If two DFSMS MANAGE commands are executed concurrently against the same storage group, the second DFSMS MANAGE command fails. Also, if one DFSMS MANAGE command and one DFSMS CONVERT command are executed concurrently against the same storage group, the second command fails. The first command executed in both instances continues processing.
- During DFSMS MANAGE command processing, DFSMS/VM can migrate files. If a file's management class indicates migration to ML2, but ML2 is not specified in the control file, the file is migrated to ML1.
- The DFSMS MANAGE command can use significant processing time to complete. It is recommended to run this command during off-shift hours.
- The DFSMS MANAGE command can be cancelled by issuing the DFSMS DISCARD command or the ISMF DISCARD command. The request ID, displayed when the MANAGE command is issued, is required when using the DFSMS DISCARD command. The DFSMS QUERY command or the ISMF QUERY command can be used to obtain the request ID.
- The WAIT option can be used in EXECs that issue multiple MANAGE commands to limit the number of MANAGES that are active. This may be useful for installations that need to do manages but have other applications running that they don't want to impact.

Example

To erase expired files, migrate files in storage group 4 using the threshold levels set in the control file and wait until command processing completes. Enter:

```
DFSMS MANAGE STORGROUP 4 (THRESHOLD WAIT
```

Figure 84 on page 149 shows an example of the output from the DFSMS MANAGE command.

```
DFSMS Function Level 221                02/26/02  09:02:41
DFSMS MANAGE with NOWAIT, request identifier 120

DFSMS MANAGE processing started for storage group 4 in file pool VMSYSU

FSMDSM3141I File STATUS LIST3820 VMSYSU:BELL.TEST. was erased
FSMDSM3141I File WEEK1 MODULE VMSYSU:BELL.TEST. was erased
FSMDSM3136I Migration of file TESTA1 EXEC VMSYSU:BELL.TEST. not allowed by FSMMECHK exit,

DFSMS MANAGE STORGRP command summary:

  High Threshold: 80  Low Threshold: 30
  Storage group percent utilization before MANAGE: 95
  Storage group percent utilization after MANAGE: 50
  48110 blocks available in primary storage group
  24953 blocks available in ML1
  2 files and directories converted
  0 files and directories could not be converted
  2 files occupying 2 4K blocks erased
  50 files eligible to be migrated
  43 files occupying 3000 4K blocks have been migrated from primary storage to ML1
  7 files occupying 276 4K blocks have been migrated from primary storage to ML2
  24 files occupying 1478 4K blocks have been moved from ML1 to ML2

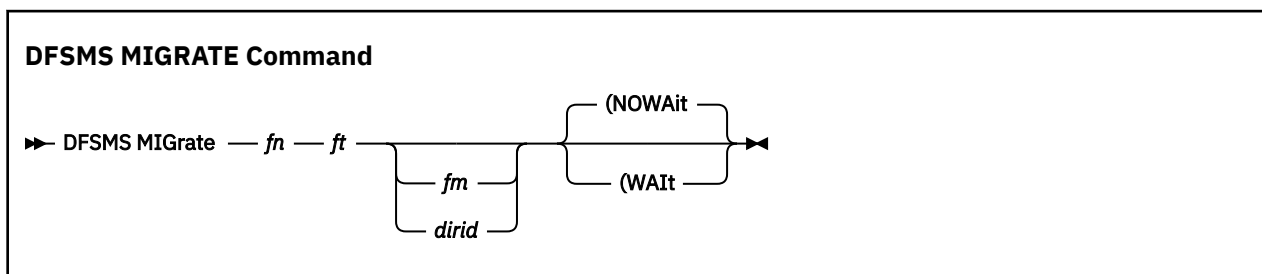
DFSMS MANAGE with NOWAIT completed with no errors
```

Figure 84. Example of the DFSMS MANAGE Command Output

DFSMS MIGRATE

The DFSMS MIGRATE command causes files to move from primary storage to secondary storage. Because the file ID can contain pattern-matching characters, this command has the capability of migrating multiple files.

The command syntax is:



Operands

fn ft

are the standard file name and file type for CMS files, each consisting of 1–8 characters. In addition pattern-matching characters (*) and % can be used as part of the file name and file type.

fm

indicates a letter between **A** and **Z** that specifies an accessed SFS directory. The *+fm.ni[.ni+1... ni+7]* forms or the *-fm[.ni.ni+1... ni+7]* forms can also be used. For additional information about SFS naming conventions, refer to the *z/VM: CMS Commands and Utilities Reference*. The file mode number is optional. A file mode of asterisk (*) is not accepted. If a file mode or directory identifier is not specified, the SFS directory accessed as file mode A is used.

dirid

is the name of the SFS directory in which the file to be processed is located. If not specified, the SFS directory accessed as file mode A is searched.

Options

NOWAIT

is an optional parameter indicating that the command issuer does not want to wait until completion of DFSMS MIGRATE command processing. Control is returned to the command issuer when the command is received by DFSMS/VM. The NOWAIT parameter is the default and cannot be used with the WAIT parameter.

WAIT

is an optional parameter indicating that the command issuer wants to wait until completion of DFSMS MIGRATE command processing. Control is not returned to the command issuer until command processing is complete. The WAIT parameter cannot be used with the NOWAIT parameter.

Usage Notes

- The user must have read/write authority to a file in order to migrate it.
- If a file's SPACE MANAGEMENT TECHNIQUE management class attribute is NONE, the file's management class prohibits migration, or if the file has an invalid management class, the file is not eligible for migration.
- If a file's management class indicates migration to ML2, but ML2 is not specified in the control file, the file is migrated to ML1.
- The pattern-matching characters * and % are allowed for file names and file types. The * character matches any number of characters. The % character matches only one character.
- Files in DIRCONTROL directories, empty files, external objects, or CMS minidisk files cannot be migrated.

- If an alias is specified when issuing the DFSMS MIGRATE command, the base file is migrated.
- The following are some of the various ways to see which files are migrated:
 - The CMS LISTFILE command with the SHARE option provides a list of files. An asterisk (*) displayed after the type field indicates the file is migrated. For additional information about the LISTFILE command, refer to the [z/VM: CMS Commands and Utilities Reference](#).
 - The CMS FILELIST command with the SHARE option provides a list of files. An asterisk (*) displayed after the type field indicates the file is migrated. For additional information about the FILELIST command, refer to the [z/VM: CMS Commands and Utilities Reference](#).
 - The DFSMS REPORT FILESPACE command generates a report of migrated files.

Example

To migrate any files with a file name of WEEKLY, any file type, and then not wait until migrate processing completes, enter:

```
DFSMS MIGRATE WEEKLY * (NOWAIT
```

Figure 85 on page 151 shows an example of the output from the DFSMS MIGRATE command.

```
DFSMS Function Level 221                02/16/02  16:50:39
DFSMS MIGRATE with NOWAIT, request identifier 57

DFSMS MIGRATE processing started for WEEKLY * VMSYSU:USER01.

    7 files matched the fileid specified
    6 files were eligible for processing
    2 files migrated to migration level 1
    4 files migrated to migration level 2

DFSMS MIGRATE with NOWAIT completed with no errors
```

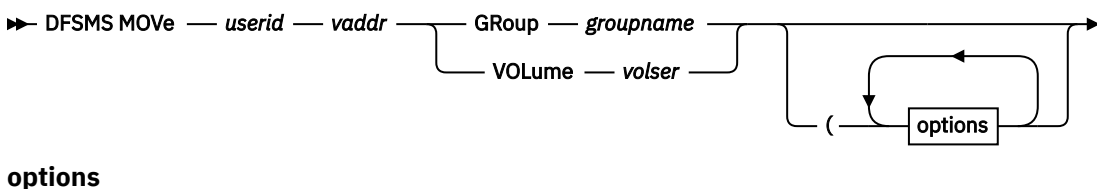
Figure 85. Example of the DFSMS MIGRATE Command Output

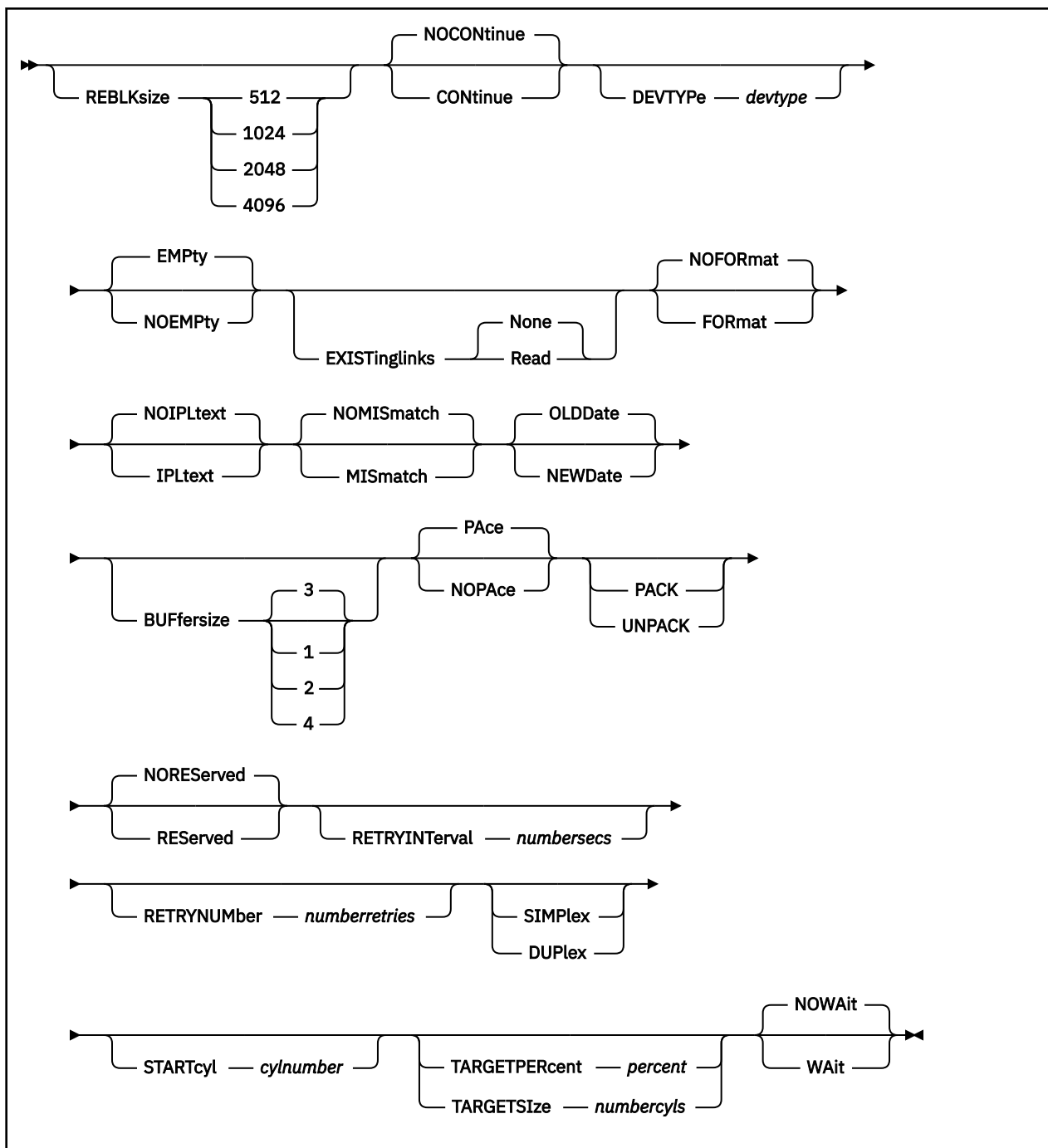
DFSMS MOVE

The DFSMS MOVE command copies a single source minidisk to a targeted destination. DFSMS MOVE processing also updates the CP directory and automatically adjusts the sizes of the allocation maps on the target disk. DFSMS MOVE processing occurs in a DFSMS minidisk server virtual machine opposed to DFSMS COPY processing that takes place in the user's virtual machine.

The command syntax is:

DFSMS MOVE Command





Operands

userid

is the user ID of the owner of the source minidisk.

vaddr

is the virtual address of the source minidisk. This virtual address must be 1–4 hex digits.

GROUP *groupname*

is the name of the group from which the target minidisk is selected. Groups specified during MOVE operations must be defined to your directory maintenance facility (for example, groups are defined in DIRMAINT through the DIRMAINT extent control file). The GROUP keyword cannot be used with the VOLUME keyword. One of these two keywords must be specified.

VOLume volser

is the volume serial name of the DASD volume where the target minidisk is located. The VOLUME keyword cannot be used with the GROUP keyword. One of these two keywords must be specified.

Options**REBLKsize**

specifies the block size of the target minidisk. The default is the same block size as the source minidisk. The block sizes to choose from are:

- 512
- 1024
- 2048
- 4096

NOCONTinue

stops minidisk moving if a permanent I/O error occurs during moving. The NOCONTINUE parameter is the default and cannot be used with the CONTINUE parameter.

CONTinue

allows minidisk moving to continue even if a permanent I/O error occurs during moving. The CONTINUE parameter cannot be used with the NOCONTINUE parameter.

DEVTYPE devtype

specifies the device type of the target volume. This parameter must be specified if the STARTCYL parameter is used and is only valid when the VOLUME keyword is used. Valid device types are:

- 3380
- 3390
- 9336
- FB-512

EMPTy

allows minidisks that are empty to be moved. The EMPTY parameter is the default and cannot be used with the NOEMPTY parameter.

NOEMPTy

prevents minidisks that are empty from being moved. The NOEMPTY parameter cannot be used with the EMPTY parameter.

EXISTinglinks

specifies link conditions for a minidisk move. This reflects the level of sharing to be allowed on the source minidisk during the move operation.

None

The source minidisk is linked in Exclusive Write (EW) mode. Existing links are not allowed when the move is initiated, and no links can be established during the move. This is the default.

Read

The source minidisk is linked in Shared Write (SW) mode. Any read links except Exclusive Read (ER) and Shared Read (SR) can exist when the move is initiated and can be established during the move.

NOFORmat

specifies that no format of the source minidisk is to be done. The source minidisk is left as it is after the move operation. The NOFORMAT parameter is the default and cannot be used with the FORMAT parameter.

FORmat

specifies to format the source minidisk to binary zeroes after the move operation is complete. The FORMAT parameter cannot be used with the NOFORMAT parameter.

NOIPLtext

specifies that the source minidisk is not moved if it contains IPL text. The NOIPLTEXT parameter is the default and cannot be used with the IPLTEXT parameter. See [Usage Notes](#) about control options.

IPLtext

specifies to move the source minidisk even if it contains IPL text. If IPL text is present when the minidisk is moved, you have to recreate IPL text on the target minidisk after the move is complete. The IPLTEXT parameter cannot be used with the NOIPLTEXT parameter. See [Usage Notes](#) about control options.

NOMISmatch

specifies that the source minidisk is not moved if there is a CP/CMS size mismatch. The NOMISMATCH parameter is the default and cannot be used with the MISMATCH parameter. See [Usage Notes](#) about control options.

MISmatch

specifies to move the source minidisk even if there is a CP/CMS size mismatch. CP/CMS size mismatches occur when the FORMAT with RECOMP command is used to reserve space (from CMS) on the minidisk. Moving a minidisk with a CP/CMS size mismatch can result in truncated data when the minidisk is moved. The MISMATCH parameter cannot be used with the NOMISMATCH parameter. See [Usage Notes](#) about control options.

OLDDate

specifies that the creation date of the source minidisk is the creation date in the label of the target minidisk. The OLDDATE parameter is the default and cannot be used with the NEWDATE parameter.

NEWDate

specifies that today's date is the creation date in the label of the target minidisk. The NEWDATE parameter cannot be used with the OLDDATE parameter.

BUFfersize

specifies the size of the unit of data transfer as follows:

- 1**
transfers data in units of a single track for count key data (CKD) devices.
- 2**
transfers data in units of two tracks for CKD devices.
- 3**
transfers data in units of five tracks for CKD devices. This is the default.
- 4**
transfers data in units of one cylinder for CKD devices.

Note: A maximum of 255 blocks are transferred during one I/O operation. Therefore, if the requested REBLKSIZE contains greater than 255 blocks, multiple I/O operations are required. Use options 1, 2, or 3 to move block sizes of 2KB or smaller on either 3380 or 3390 devices.

PAce

specifies that a two-second delay between each unit of input data. This is recommended during prime shift to prevent impacting the system. This PACE parameter is the default and cannot be used with the NOPACE parameter.

NOPAce

specifies no delays are made during the minidisk move. The NOPACE parameter cannot be used with the PACE parameter.

PACK

specifies that all files on the minidisk are packed on the target minidisk, during the move operation, minimizing the allocated space. Using this parameter prohibits use of the high-speed data mover. When the PACK or UNPACK parameter is not specified then the high-speed data mover is invoked during move processing.

UNPACK

specifies that all files on the minidisk are unpacked on the target minidisk for ease of use. Using this value prohibits use of the high-speed data mover. When the PACK or UNPACK parameter is not specified then the high-speed data mover is invoked during move processing.

NOREReserved

prohibits the move operation if the source minidisk is a CMS reserved minidisk. The NORESERVED parameter is the default and cannot be used with the RESERVED parameter. See [Usage Notes](#) about control options.

REServed

specifies that the move operation occurs even if the source minidisk is a CMS reserved minidisk. The RESERVED parameter cannot be used with the NORESERVED parameter. See [Usage Notes](#) about control options.

RETRYINterval *numbersecs*

is the number of seconds for DFSMS/VM to wait between successive link attempts. If not specified, the default is 10 seconds.

RETRYNUMber *numberretries*

is the number of retries to link the source minidisk if the first link fails before failing the MOVE operation. If not specified, the default is two retries.

SIMPLex

specifies that the move request only begins if the target minidisk resides on a volume where duplexing is not available. If neither SIMPLEX or DUPLEX is specified, the move request proceeds without checking the duplex status of the target minidisk.

DUPlex

specifies that the move request only begins if the target minidisk resides on a duplex volume. If neither SIMPLEX or DUPLEX is specified, the move request proceeds without checking the duplex status of the target minidisk.

STARTcyl *cylnumber*

specifies the starting cylinder location of the target minidisk on the target volume. This parameter must be specified if the DEVTYPE parameter is used and is only valid when the VOLUME keyword is used.

TARGETPERcent *percent*

is the size of the target minidisk as a percentage of the source minidisk size. The default is 100 percent of the source minidisk, that is, the target minidisk is the same size as the source minidisk. This keyword is mutually exclusive with the TARGETSIZE keyword.

TARGETSize *numbercyls*

is the size of the target minidisk in number of cylinders. This keyword is mutually exclusive with the TARGETPERCENT keyword.

NOWait

is an optional parameter indicating that the command issuer does not want to wait until completion of DFSMS MOVE command processing. Control is returned to the command issuer when the command is received by DFSMS/VM. The NOWAIT parameter is the default and cannot be used with the WAIT parameter.

WAIT

is an optional parameter indicating that the command issuer wants to wait until completion of DFSMS MOVE command processing. Control is not returned to the command issuer until command processing is complete. The WAIT parameter cannot be used with the NOWAIT parameter.

Restrictions

DFSMS MOVE has the following restrictions:

- The target minidisk must hold at least as many CMS blocks as the source minidisk.
- On a minidisk processed by CMS FORMAT with the RECOMP option, data outside of the CMS-formatted portion of the minidisk is not copied to the target minidisk.

- The specified block size for the target minidisk must be supported on the target device type.
- If reblocking is requested, then CMS FORMAT and CMS COPYFILE are invoked instead of DFSMS COPY. In addition, CMS FORMAT and CMS COPYFILE are invoked if PACK or UNPACK is specified, or the source minidisk is empty.
- CMS must have enough virtual storage under 16 MB available to access the source and target disks.

Note: Because the CMS file system requires file status and control information to reside below 16 MB in virtual storage, there is a practical limitation on the size of CMS minidisks. As a minidisk increases in size, or more files reside on the disk, the amount of virtual storage associated with the disk for CMS file system status and control increases in storage below 16 MB. The current ECKD DASD limitation is 32767 cylinders for a 3390 disk on an IBM TotalStorage DASD subsystem, or about 22 GB of data. IBM suggests that customers defining FBA SCSI disks for use by CMS should set a practical limit of about 22 GB. If larger disks are defined, they should be limited to contain very few files, or the CMS file system may not be able to obtain enough virtual storage below 16 MB to format or access those disks. For more information, see the ACCESS command in the *z/VM: CMS Commands and Utilities Reference*. The maximum size for FBA SCSI disks supported for use by CMS or GCS is 381 GB.

Usage Notes

- After a successful minidisk move, a message is issued stating the new volume and starting cylinder of the new minidisk.
- The DFSMS/VM control file contains minidisk options permitting the installation to control the MOVE of a minidisk with certain attributes. These attributes are:
 - Minidisk contains IPL text
 - CP and CMS cylinder counts differ for the minidisk
 - Minidisk is reserved

If the user specified option conflicts with the DFSMS control option, the MOVE fails.

- When MOVE command completes, the user receives a reader file with the status of the move operation.

Example

The source minidisk owned by user ID *bill* with a virtual address of *EFB0* must be moved to the *payroll* volume. The required block size of the new minidisk is *1024* and the command issuer needs to wait for move processing completion. Enter:

```
DFSMS MOVE bill EFB0 VOLUME payroll (BLKSIZE 1024 WAIT
```

Figure 86 on page 156 shows an example of the output from the DFSMS MOVE command.

```
DFSMS Function Level 221                      01/27/02   13:01:41
DFSMS MOVE with NOWAIT, request identifier 3

The following options were specified for the Source Minidisk:
  IPLTEXT Present: N      Size Mismatch: N      Reserved Minidisk: N
  Empty Minidisk: Y      Format Source: N      Existing Links: 1
  Link Retries: 2       Link Interval: 000010

The following options were specified for the Target Minidisk:
  Group:                Volume: USRSRV
  Minidisk Type: 1      Creation Date: OLDDATE  Minidisk Size: 100%
  Pace Level: 2         Buffer Size: 3          Continue On Error: N
  Pack Option: 3        Reblock Files: N        Reblock Size:

FSMEMD4000I DFSMS MOVE request identifier 3 (VMTEST6 191) succeeded

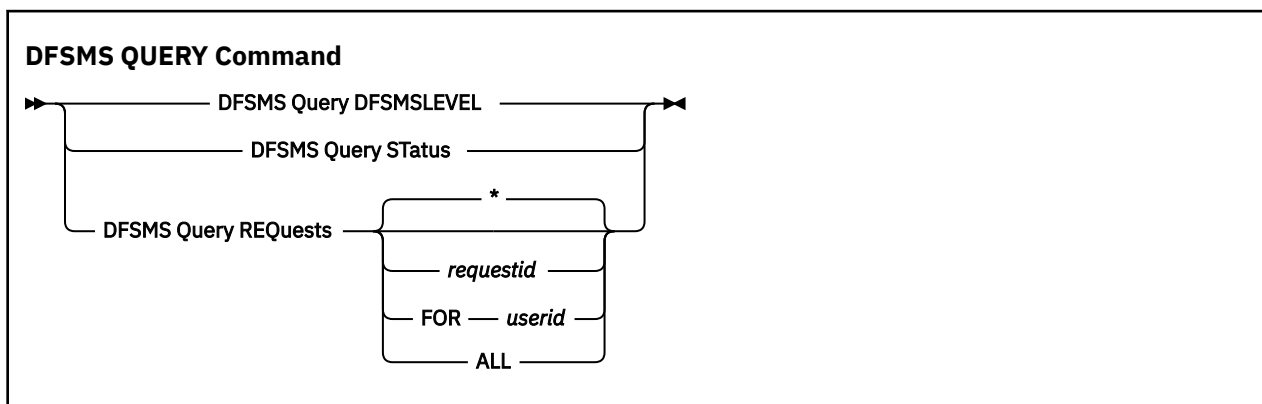
DFSMS MOVE with NOWAIT completed
```

Figure 86. Example of the DFSMS MOVE Command Output

DFSMS QUERY

The DFSMS QUERY command provides information about the level of DFSMS/VM performing space management, the status of various master and server machines, and the status of command requests. The type of information you receive is dependent on the keyword you select to issue with the DFSMS QUERY command. The keywords are DFSMSLEVEL, STATUS, and REQUESTS.

The command syntax is:



Operands

DFSMSLEVEL

provides the DFSMS/VM function level performing storage management processing on your system.

STATUS

provides the status of the DFSMS master and server machines, and the status of the migration level 1 (ML1) repository and the migration level 2 (ML2) repository (if space management is enabled). Status includes which virtual machines are logged on and the number of requests queued for processing.

REQUESTS

provides the status of all work requests accepted by DFSMS/VM. You can query a single request, all requests for a single user, or all requests to which you are authorized.

Options for DFSMS QUERY REQUESTS

specifies that all requests for the command issuer be queried. This is the default.

requestid

identifies the request for which the processing status is queried.

FOR userid

specifies that all requests for the given user ID be queried. The command fails if the command issuer does not have authorization to query the requests of the given user.

ALL

specifies that all requests that the command issuer has authorization for are queried.

Usage Notes

- The output that is generated is sent back to the issuer and is displayed on the user's console.
- The messages returned by the DFSMS QUERY STATUS command indicate:
 - Whether DFSMS/VM is running or stopping
 - How many outstanding requests are currently in the system
 - The status of the DFSMS servers
 - The status of the ML1 and ML2 servers

RECALL

- The status of the minidisk servers
- DFSMS QUERY REQUESTS can obtain the status of the following commands:
 - ALTER
 - CHECK (minidisk)
 - CONVERT
 - DELETE
 - MANAGE
 - MIGRATE
 - MOVE (minidisk)
 - RECALL (command recall only)
 - REPORT

Example

To display the status of the DFSMS master and servers, enter:

```
DFSMS QUERY STATUS
```

Figure 87 on page 158 shows an example of the output from the DFSMS QUERY command.

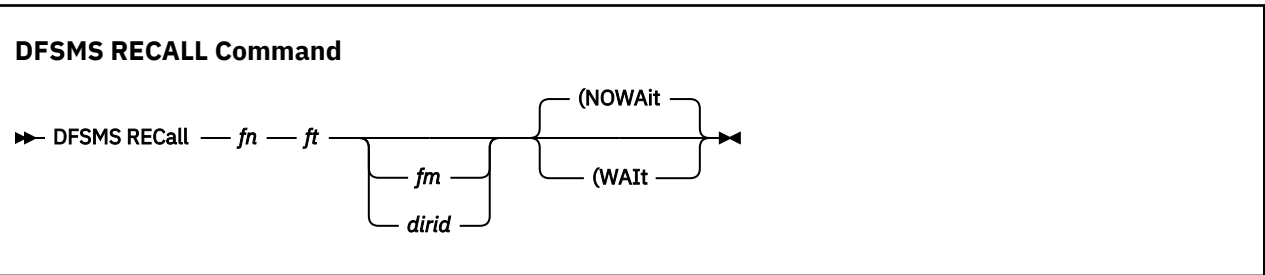
```
DFSMS Q STATUS
FSMDPS3161I DFSMS master is running
FSMDPS3162I DFSMS has 1 outstanding requests
FSMDPS3200I DFSMS server SMSSRV02 has 10 tasks available for work and 0 tasks busy
FSMDPS3200I DFSMS server SMSSRV01 has 10 tasks available for work and 0 tasks busy
FSMDPS3200I DFSMS server SMSSRV03 has 10 tasks available for work and 0 tasks busy
FSMDPS3114I Minidisk server DGTSRV01 is waiting for work
FSMDPS3114I Minidisk server DGTSRV02 is waiting for work
FSMDPS3114I Minidisk server DGTSRV03 is waiting for work
FSMDPS3114I ML1 file pool MIGLEV05 is available
FSMDPS3108E ML2 server DSSERV is unavailable
```

Figure 87. Example of the DFSMS QUERY Command Output

DFSMS RECALL

The DFSMS RECALL command causes files to move from secondary storage to primary storage. The file pool catalog entry for the recalled file is updated to indicate that the file is no longer migrated. Because the file ID can contain pattern-matching characters, this command can recall multiple files.

The command syntax is:



Operands

fn ft

are the standard file name and file type for CMS files, each being 1–8 characters. In addition pattern-matching characters (* and %) can be used as part of the file name and file type.

fm

indicates a letter between **A** and **Z** that specifies an accessed SFS directory. The *+fm.ni[.ni+1... ni+7]* forms or the *-fm[.ni.ni+1... ni+7]* forms can also be used. For additional information about SFS naming conventions, refer to the *z/VM: CMS Commands and Utilities Reference*. The file mode number is optional. A file mode of asterisk (*) is not accepted, and if a file mode or directory identifier is not specified, only the directory accessed as file mode A is searched.

dirid

is the name of the SFS directory in which the file to be processed is located. If a file mode is not specified, the SFS directory accessed as file mode A is searched.

Options**NOWAIT**

is an optional parameter indicating that the command issuer does not want to wait until completion of DFSMS RECALL command processing. Control is returned to the command issuer when the command is received by DFSMS/VM. The NOWAIT parameter is the default and cannot be used with the WAIT parameter.

WAIT

is an optional parameter indicating that the command issuer wants to wait until completion of DFSMS RECALL command processing. Control is not returned to the command issuer until command processing is complete. The WAIT parameter cannot be used with the NOWAIT parameter.

Usage Notes

- When the files have been recalled, the user receives a reader file with the status of each file.
- You must have at least read authority to recall the file.
- The pattern-matching characters ***** and **%** are allowed. The ***** character matches any number of characters. The **%** character matches only one character.
- The DFSMS RECALL command can be used to recall a file even if CMS SET RECALL OFF has been set on a virtual machine.
- The WAIT option is useful for EXECs to ensure that files are recalled before starting other processing.
- ACS processing occurs for recalled files and can reject file recall.

Example

To command recall a file, MYFILE SCRIPT, from secondary storage to the SFS directory you have accessed as **A** and then wait until DFSMS RECALL command processing is complete, enter:

```
DFSMS RECALL MYFILE SCRIPT (WAIT
```

Figure 88 on page 159 shows an example of the output from the DFSMS RECALL command.

```
DFSMS Function Level 221                02/28/02   18:05:02
DFSMS RECALL with WAIT, request identifier 120

DFSMS RECALL processing started for MYFILE SCRIPT VMSYSU:VMTEST0.

    1 files matched the fileid specified
    1 files were eligible for processing
    1 files recalled from migration level 1
    0 files recalled from migration level 2

DFSMS RECALL with WAIT completed with no errors
```

Figure 88. Example of the DFSMS RECALL Command Output

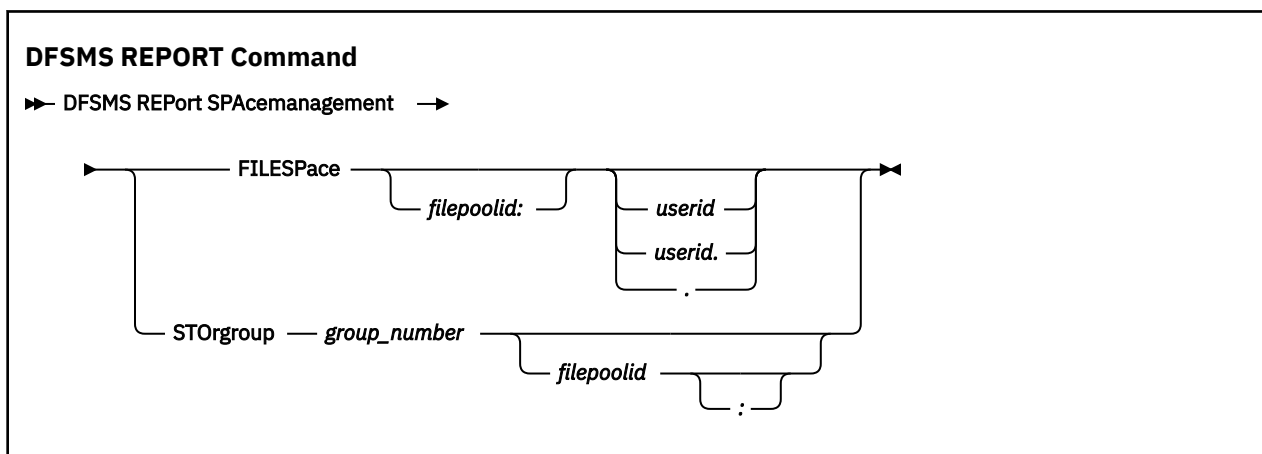
DFSMS REPORT

The DFSMS REPORT command generates a report of migrated files from either a particular file space or storage group. The reports include:

- Physical space utilization for primary storage, migration level 1 (ML1) storage, and migration level 2 (ML2) storage
- Space savings due to compaction of migrated data
- Out-of-sync conditions between SFS and DFSMS/VM catalogs

The report is returned to the command issuer as a reader file.

The command syntax is:



Operands for the FILESPACE Keyword

filepoolid:

specifies the file pool containing the file space for which a report is generated. If not specified, the user's current file pool ID is used. If this operand is used, be sure to include the colon (:).

userid

specifies the user ID which, when combined with the file pool, identifies the file space for which a report is generated. If not specified, the user ID of the command issuer is used.

Operands for the STORGROUP Keyword

group_number

identifies the storage group for which a report is generated. Any defined storage group except storage group 1 can be specified in this command. If storage group 1 is specified, an error results and DFSMS REPORT command processing terminates.

filepoolid

specifies the file pool containing the storage group for which a report is generated. If not specified, the user's current file pool ID is used. An optional colon is allowed after the file pool ID.

Usage Notes

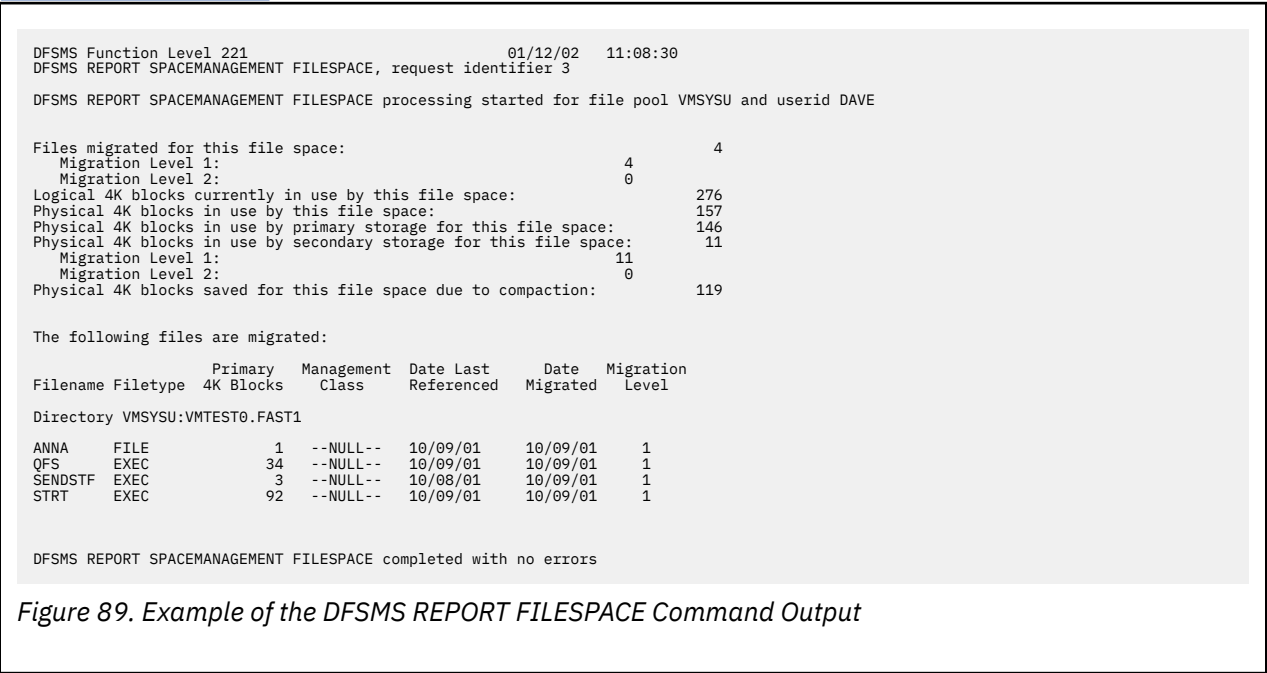
- You can use this command to determine if an out-of-sync condition exists between SFS and DFSMS catalogs. Use the DFSMS DELETE command to resolve any out-of-sync conditions.
- To issue a report against another user, you must be both a DFSMS/VM administrator and an SFS administrator.

Example

To generate a file space report for user ID *DAVE* in the *VMSYSU* file pool, enter:

```
DFSMS REPORT SPACEMANAGEMENT FILESPACE VMSYSU:DAVE.
```

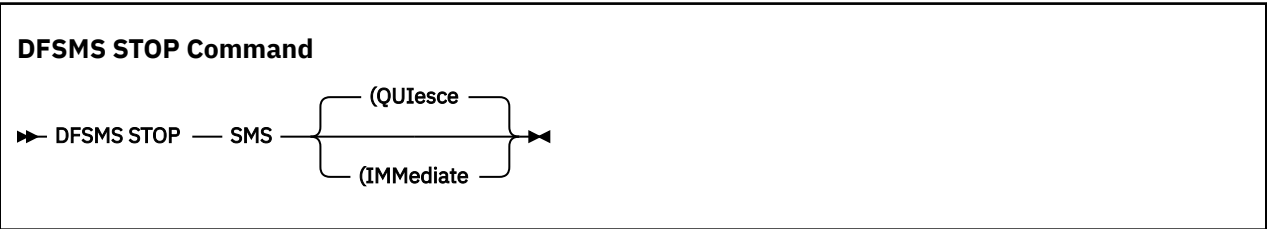
Figure 89 on page 161 shows an example of the output from the DFSMS REPORT command.



DFSMS STOP

The DFSMS STOP command is used to shut down the DFSMS master and server and minidisk server machines. The DFSMS STOP command cannot be used to stop DFSMS/VM on another system.

The command syntax is:



Operand

SMS
specifies that processing will stop on the DFSMS master, server, and minidisk server virtual machines.

Options

QUIesce
indicates that a quiesced shutdown is requested.

IMMediate
indicates that an immediate shutdown is requested.

Usage Notes

- When an quiesced shutdown is requested, DFSMS/VM finishes processing all accepted requests and logs off the DFSMS master and servers

- When an immediate shutdown is requested, DFSMS/VM:
 - Rejects all new requests from users.
 - Completes all *active* tasks (the smallest unit of work that DFSMS/VM can perform). If DFSMS/VM receives a request to migrate 10 files, the migration of *each* file is a single task.
 - Cancels all requests that have not started. See Chapter 11, “Starting and Stopping DFSMS/VM Processing,” on page 111 for additional information.
 - Cancels all active MANAGE and CONVERT requests.
 - Notifies users affected by the shutdown on the status of their requests.
 - Logs off the DFSMS master and servers when all requests are finished.

See Chapter 11, “Starting and Stopping DFSMS/VM Processing,” on page 111 for additional information.

Example

To immediately shut down the DFSMS master and servers, you would enter:

```
DFSMS STOP SMS (IMMEDIATE
```

Figure 90 on page 162 shows an example of the output from the DFSMS STOP command.

```
FSMEPS3151I DFSMS STOP SMS command accepted. DFSMS is stopping
```

Figure 90. Example of the DFSMS STOP Command Output

Using ISMF Commands and Line Operators

ISMF includes commands and line operators that you use to perform a variety of tasks. You enter commands on the command line of all ISMF panels, and you enter line operators in the left column of the ISMF list panels. You can use ISMF commands to issue a request for a single entry in a list. ISMF also provides list commands you can use to issue requests against all the displayable entries in a list.

Most commands and line operators are processed interactively. That is, ISMF performs the task immediately.

When you issue certain ISMF commands, such as the MIGRATE command, you are issuing a request to perform a task. These requests are queued for processing and performed in sequential order as server machines become available.

The following sections contain additional information about using ISMF commands and line operators. You can also display detailed information about each command and line operator by using the ISMF help facility. You can enter the full name of a command or line operator or use an abbreviation.

ISMF Commands

ISMF is comprised of a subset of ISPF commands and includes commands you can use to return to previous panels (END, RETURN, CANCEL), to scroll up and down through list entries (UP, DOWN), to go to the top or bottom of a list (TOP, BOTTOM), or to move columns of text left and right on a panel (LEFT, RIGHT). The remaining ISMF commands can be used on most ISMF panels to work with and modify lists.

Important

Any userid issuing a DFSMS command or using the ISMF interface (as well as issuing a DFSMSRM command or using the CSL interface), must have an IUCV ANY statement in its directory. See [z/VM: CP Planning and Administration](#) for more information on including this statement in a user directory.

ISMF Command Action

ACTivate

Makes a source configuration the active configuration for use by DFSMS/VM. This command is only applicable in the Configuration Application.

ALTER FILESPace

Alters the management class of a top-level directory without converting the entire storage group. ALTER FILESPACE is only applicable in the file application.

BOTtom

Causes a scroll to the end of the data.

CLEar

From a list panel, the CLEAR command sets the line operator column to blanks, resets the line operator history, and removes messages if any are currently displayed. From an application selection entry panel, the CLEAR command sets entry fields to default values or leaves the field blank if no default is defined. For the entry panel, the CLEAR command uses the following parameters:

CLEar PAGE

Clears the current page of the entry panel

CLEar PAGEx

Clears the page of the entry panel specified by x

CLEar ALL

Clears all pages of the entry panel

CP/CMS

CP or CMS commands can be entered while in ISMF by entering CP or CMS on the command line followed by the command you wish to enter. Commands which are prefixed with CP or CMS are invoked directly by ISPF, and, as such, ISMF will not issue any messages indicating the status of any CP or CMS commands.

CURSOR

Moves the cursor to the first input field on line two, which is normally the OPTION or COMMAND field.

DISCard

Cancels DFSMS commands that have not started and cancels DFSMS MANAGE or DFSMS CONVERT commands in-process. This is available on all panels except the ISMF Error Table Display panel.

Down

Causes a scroll toward the bottom of the data.

END

Cancels any entries you have made on the current panel, cancels the current task, and returns to the previous panel. However, on the Base Configuration Define, Management Class Define, and Profile Application panels, the END command saves the values which completes the current task and returns to the previous panel.

ERTb

Displays the ISMF Error Table Display panel. This is available on all panels except the ISMF Error Table Display panel itself.

EXECUTE

The EXECUTE command is the only method available for processing non-ISMF commands against list panel entries. For example:

```
COMMAND ==> EXECUTE copyfile / = = b
```

allows controlled list processing for the named command against all displayed list entries and allows CP/CMS to process the command and issue messages.

Note: Preceding a command with CP or CMS causes that command to be processed in that environment without any list data being passed. The list data is available to the EXECUTE command in the ISPF shared variable pool, or list data can be passed as parameters depending on how the command is invoked. EXECUTE is only applicable on ISMF list panels.

FILter

Filters a list to contain only those list items that meet the criteria you specify. On a list panel, you can enter FILTER CLEAR to clear all filter criteria and redisplay the unfiltered list. FILTER is only applicable on list panels.

FIND

Scrolls left or right to requested column tag. FIND is only applicable on ISMF list panels.

HELP

Requests the long error message help and the help panels associated with the current panel.

Left

Causes a scroll toward the left margin of the data.

MIGrate

Migrates all files in the file list. MIGRATE is only applicable on file list panels.

MOve

Requests the move of all displayable entries in a minidisk list.

Profile

Selects the Profile Application. Available on all panels except the ISMF Primary Option menu, the ISMF Profile Option menu, the Logging Control Entry panel, and the ISMF Error Table Display panel.

Query

Obtains status information about a request. Available on all panels except the ISMF Error Table Display panel and the Query Display panel.

RECall

Recalls all files in the file list. RECALL is only applicable on file list panels.

REFresh

Regenerates a list using your original selection criteria or using the contents of a saved list. REFRESH is only applicable on list panels.

RESHow

Reshows entries you have hidden using the HIDE line operator. RESHOW is only applicable on list panels.

RETRIEVE

Regenerates the previous command that was entered.

RETURN

Returns to the primary option menu used to get to the current panel. If you are using the Minidisk or List Applications when you use the RETURN command, you return to the ISMF Primary Option menu. If you are using the Profile Application, you return to the ISMF Profile Option menu.

RIght

Causes a scroll toward the right margin of the data.

SAve

Saves a list containing the entries that are not hidden or filtered. SAVE is only applicable on ISMF list panels.

SOrt

Sorts a list in the order you specify. SORT is only applicable on ISMF list panels.

SPLIT

Causes split screen mode to be entered, or changes the location of the split line.

STOP

Shuts down the DFSMS master and server machines.

SWAP

Moves the cursor to the screen location that it was previously positioned on the other logical screen of a split-screen pair.

Top

Causes a scroll to the beginning of the data.

Up

Causes a scroll toward the top of the data.

VALIDATE

Validates a source configuration. VALIDATE is only applicable on Configuration Application panels.

ISMF Command Status

When you issue ISMF commands, most functions are processed synchronously giving you immediate feedback. For example, when you issue a REFRESH command on the minidisk panel command line, ISMF issues a message letting you know your request is being processed and then refreshes the panel. When you issue an ERTB command, ISMF displays the ISMF Error Display panel.

If there is a problem with any command, ISMF issues a short message that describes the error in the upper right-hand corner of the panel. You can enter HELP or press PF1 to display the long message that provides more detail about the error. See [“Online HELP for DFSMS Commands” on page 133](#) for details on displaying ISMF messages.

When you issue a list command that requests processing for a whole list of requests, the messages that appear are summary messages. Use the MESSAGE line operator to display the diagnostic for each entry in the list.

When you issue an user extension (for example, PRNTMINI) as a list command by means of the EXECUTE feature, the message set is a summary message. As with ISMF list commands, use the MESSAGE line operator to display the diagnostic message (if any) set by the user extension for each entry in the list.

When you issue a command against all items in a list, in addition to issuing messages, ISMF uses the line operator column to give you information about each entry in the list.

ISMF Line Operators

ISMF includes line operators that you can use to perform tasks or issue requests against individual list entries in a list.

In the Minidisk Application:

CHeck

Checks the integrity of a CMS minidisk. CHECK requests are queued for processing.

DISCard

Discards a request that has not been processed.

Hide

Temporarily hides an entry or group of entries from a list. **Hnnnnn** causes a number (**n**) of list entries to be hidden, starting with the entry against which H and a number from 1 to 99999 is entered. Enter H and the number of entries you want to hide (for example, H9) in column one of the first list entry you want to hide. Do not add a blank space between H and the number of entries you want to hide. ISMF interprets the blank as the end of the HIDE line operator and hides only the first entry.

MESSAGE

Displays message text related to the last operation performed on a particular list entry.

MOve

Moves a CMS minidisk from one location to another. MOVE requests are queued for processing.

Query

Obtains status information for a single minidisk for which a MOVE or CHECK request has been issued.

In the List Application:

ERase

Erases saved minidisk list, file list, or management class lists.

Hide

Temporarily hides an entry or group of entries from a list. **Hnnnnnn** causes a number (**n**) of list entries to be hidden, starting with the entry against which H and a number from 1 to 99999 is entered. Enter H and the number of entries you want to hide (for example, H9) in column one of the first list entry you want to hide. Do not add a blank space between H and the number of entries you want to hide. ISMF interprets the blank as the end of the HIDE line operator and hides only the first entry.

LIsT

Allows you to display any saved list.

MESSAGE

Displays message text related to the last operation performed on a particular list entry.

In the File Application:**ALter**

Associates a new management class with a file or directory. Use the ALTER FILESPACE command to alter the management class of a top level directory.

Browse

Views the contents of a file.

Edit

Edits a file.

FIlelist

Generates an additional file list.

LIsTfile

Generates an additional file list.

Hide

Temporarily hides an entry or group of entries from a list. **Hnnnnnn** causes a number (**n**) of list entries to be hidden, starting with the entry against which H and a number from 1 to 99999 is entered. Enter H and the number of entries you want to hide (for example, H9) in column one of the first list entry you want to hide. Do not add a blank space between H and the number of entries you want to hide. ISMF interprets the blank as the end of the HIDE line operator and hides only the first entry.

MESSAGE

Displays message text related to the last operation performed on a particular list entry.

MIGrate

Allows a file migration from the line operator field.

RECall

Recalls a file from secondary storage.

Xedit

Invokes the XEDIT editor for the selected file.

In the Management Class Application:**ALter**

Changes the expiration and migration attributes of a management class.

COpy

Copies the attributes of one management class to another management class in the same or a different source configuration file.

DElete

Deletes a management class for the configuration.

DISPlay

Displays the expiration and migration attributes for the specified management class.

Hide

Temporarily hides an entry or group of entries from a list. **Hnnnnnn** causes a number (**n**) of list entries to be hidden, starting with the entry against which H and a number from 1 to 99999 is entered. Enter

H and the number of entries you want to hide (for example, H9) in column one of the first list entry you want to hide. Do not add a blank space between H and the number of entries you want to hide. ISMF interprets the blank as the end of the HIDE line operator and hides only the first entry.

MESSAGE

Displays message text related to the last operation performed on a particular list entry.

More On Line Operators

This section contains the additional information about line operators. Topics include:

- Repeating a line operator
- Using last-use mode
- Obtaining line operator history

You can use the same line operator against more than one list entry at a time.

To repeat a line operator:

1. Type the line operator name or abbreviation in the line operator field next to the first entry against which you want to perform an operation.
2. Type an equal sign (=) in the line operator field next to any number of subsequent entries in the list against which you want to perform the same operation.

Figure 91 on page 167 shows sample entries for issuing move requests for multiple minidisks.

DGTNLML1
COMMAND ==>>>

MINIDISK LIST PANEL

SCROLL ==>> HALF

Entries 1-14 of 16

Data Columns 4-7 of 13

ENTER LINE OPERATORS BELOW:

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	MDISK START	MDISK END	MDISK SIZE	MDISK GAP
---(1)---	---(2)---	---(3)---	---(4)---	---(5)---	---(6)---	---(7)---
MOVE	MAINTDEV	1A1	30	129	100	0
=	MAINTDEV	19F	130	229	100	0
=	MAINTDEV	120	230	479	250	0
=	MAINTDEV	19A	480	879	400	0
=	MAINTDEV	BC1	880	909	30	496
=	MONWRITE	191	1406	1605	200	0
=	CSERV0X0	191	1606	1610	5	54
	RTMTSTA	191	1665	1694	30	0
	RTMTSTB	191	1695	1704	10	0
	RTMTSTC	191	1705	1714	10	0
	G32TST	191	1715	1744	30	165
	STUMP	191	1910	1929	20	0
	BORREG01	191	1930	1949	20	0
	BEACHMA	191	1950	1958	9	0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 91. Example of Repeating a Line Operator

When you repeat an ALTER, MOVE, CHECK, MIGRATE, or RECALL line operator, ISMF displays data entry panels for each request. If you want to repeat a line operator for multiple minidisks, but do not need to enter different options on entry panels for each request, you can use the line operator in last-use mode.

Using a Line Operator in Last-Use Mode

To enter a command in **last-use mode**, the line operator is entered followed by an equal sign in the line operator field. ISMF does not display the entry panel for the line operator. Instead, the line operator is processed with the values present on the entry panel the last time the line operator was executed. This is useful when the command issuer remembers what processing options were specified the last time a line operator was used, and they do not need to change them.

You can use a line operator in last-use mode for one entry or for multiple entries. You can also repeat a line operator in last-use mode for one or more entries.

To use a line operator in last-use mode for one entry, type the line operator name or abbreviation, followed by an equal sign (=) in the line operator field next to the entry against which you want to issue the request in last-use mode.

Figure 92 on page 168 shows a sample of using a line operator in last-use mode.

```
DGTLNML1
COMMAND ==>

MINIDISK LIST PANEL

                                SCROLL ==> HALF
                                Entries 1-14 of 16
                                Data Columns 4-7 of 13

ENTER LINE OPERATORS BELOW:
```

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	MDISK START	MDISK END	MDISK SIZE	MDISK GAP
---(1)---	---(2)---	---(3)---	---(4)---	---(5)---	---(6)---	---(7)---
MOVE=	MAINTDEV	1A1	30	129	100	0
	MAINTDEV	19F	130	229	100	0
	MAINTDEV	120	230	479	250	0
	MAINTDEV	19A	480	879	400	0
	MAINTDEV	BC1	880	909	30	496
	MONWRITE	191	1406	1605	200	0
	CSERV0X0	191	1606	1610	5	54
	RTMTSTA	191	1665	1694	30	0
	RTMTSTB	191	1695	1704	10	0
	RTMTSTC	191	1705	1714	10	0
	G32TST	191	1715	1744	30	165
	STUMP	191	1910	1929	20	0
	BORREG01	191	1930	1949	20	0
	BEACHMA	191	1950	1958	9	0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 92. Using a Line Operator in Last-Use Mode

If you have issued a previous move request, ISMF uses the options you specified for that request to process the current request.

To use last-use mode for multiple entries:

1. Type the line operator name or abbreviation, followed by an equal sign (=) in the line operator field next to the first entry against which you want to issue a request in last-use mode.
2. Enter an equal sign (=) in the line operator field next to any number of subsequent entries in the list against which you want to issue the same request in last-use mode.

Figure 93 on page 168 shows a sample of using last-use mode for multiple entries.

```
DGTLNML1
COMMAND ==>

MINIDISK LIST PANEL

                                SCROLL ==> HALF
                                Entries 1-14 of 16
                                Data Columns 4-7 of 13

ENTER LINE OPERATORS BELOW:
```

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	MDISK START	MDISK END	MDISK SIZE	MDISK GAP
---(1)---	---(2)---	---(3)---	---(4)---	---(5)---	---(6)---	---(7)---
MOVE=	MAINTDEV	1A1	30	129	100	0
=	MAINTDEV	19F	130	229	100	0
=	MAINTDEV	120	230	479	250	0
=	MAINTDEV	19A	480	879	400	0
=	MAINTDEV	BC1	880	909	30	496
=	MONWRITE	191	1406	1605	200	0
=	CSERV0X0	191	1606	1610	5	54
	RTMTSTA	191	1665	1694	30	0
	RTMTSTB	191	1695	1704	10	0
	RTMTSTC	191	1705	1714	10	0
	G32TST	191	1715	1744	30	165
=	STUMP	191	1910	1929	20	0
	BORREG01	191	1930	1949	20	0
	BEACHMA	191	1950	1958	9	0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 93. Using Last-Use Mode for Multiple Entries

ISMF uses the same move options to process a request for each entry you specify.

To repeat a line operator in last-use mode:

Enter double equal signs (==) next to the list entry against which you want to issue the request. The first equal sign indicates that the previous line operator will be repeated, and the second equal sign indicates that the line operator in last-use mode will be repeated.

Figure 94 on page 169 shows sample entries to repeat a line operator (MOVE) in last-use mode, and sample entries to repeat a line operator (CHECK) in last-use mode for more than one entry.

```

DGTNLML1
COMMAND ==>
MINIDISK LIST PANEL
                                SCROLL ==> HALF
                                Entries 1-14 of 16
                                Data Columns 4-7 of 13
ENTER LINE OPERATORS BELOW:

```

LINE OPERATOR	MDISK OWNERID	MDISK ADDR	MDISK START	MDISK END	MDISK SIZE	MDISK GAP
---(1)---	---(2)---	---(3)---	---(4)---	---(5)---	---(6)---	---(7)---
MOVE=	MAINTDEV	1A1	30	129	100	0
==	MAINTDEV	19F	130	229	100	0
	MAINTDEV	120	230	479	250	0
	MAINTDEV	19A	480	879	400	0
	MAINTDEV	BC1	880	909	30	496
	MONWRITE	191	1406	1605	200	0
	CSERV0X0	191	1606	1610	5	54
	RTMTSTA	191	1665	1694	30	0
CHECK=	RTMTSTB	191	1695	1704	10	0
=	RTMTSTC	191	1705	1714	10	0
=	G32TST	191	1715	1744	30	165
=	STUMP	191	1910	1929	20	0
	BORREG01	191	1930	1949	20	0
	BEACHMA	191	1950	1958	9	0

USE HELP COMMAND FOR HELP; USE END COMMAND TO EXIT.

Figure 94. Repeating a Line Operator in Last-Use Mode

Obtaining Line Operator History

When ISMF processes line operators, it performs the following steps to provide you with line operator history:

- Displays a short message in the top-right corner of the screen that indicates the status of the last request issued.
- Places a symbol that indicates the status of the request in the line operator field. The symbol is followed by the line operator name.
- Records a request ID in a hidden field in the list entry for MOVE and CHECK requests that are successfully accepted and queued.
- Logs processing information in the ISPF log for MOVE, CHECK, and ERASE. For move and check requests, the log information also includes the request ID.

Messages

If you enter multiple line operators or repeat a line operator, the message displayed at the top-right corner of the panel applies to the net result of all resulting requests.

To view messages related to the last request issued against other list entries, type **MESSAGE** in the line operator field next to specific list entries. You can type over the line operator history placed in this field by ISMF.

Symbols

The following table shows the symbols that can appear in the line operator field to indicate the status of a line operator you have issued.

Symbol	Example	Description
*	*MOVE	Your request has been accepted.
¬	¬MOVE	Your request <i>has not been</i> accepted, but subsequent requests from the same list are being processed.

Symbol	Example	Description
?	?MOVE	Your request <i>has not been</i> accepted, and subsequent requests from the same list <i>cannot</i> be processed.

If a not sign (¬) or question mark (?) appears in response to a line operator, you can:

- Use the MESSAGE line operator, if necessary, and the associated message help panel to obtain information about why the line operator has not been accepted or processed.
- Use the HIDE line operator to exclude an entry for which you have received a question mark (?) and to attempt to continue processing any other line operators you have entered.
- Use the CLEAR command to clear the line operator column.
- Use the SORT command to sort a list by the contents of the line operator field, arranging the list in groups by entries:
 - Against which you have not issued line operators
 - For which line operators were successfully accepted
 - For which line operators failed

Sorting the list by line operator field helps you determine which tasks are complete, and which tasks have encountered problems that you need to resolve.

For more information about using the MESSAGE line operator and Message Help panel, see “[Help Messages](#)” on page 32. For more detailed information about resolving problems, see [z/VM: DFSMS/VM Diagnosis Guide](#)

ISPF Log Information

ISMF records information about processing line operators in the ISPF log to:

- Provide a record of the tasks you perform during an ISMF session
- Record error information to help you resolve problems
- Supply an additional way to obtain a request ID

That information can be very useful. For example, if you create a new minidisk list, issue move requests, and do not save the list, you would not be able to use the list again later to obtain request IDs. If that should happen and you want to use the QUERY command for a particular request, you can check your ISPF log to find the request ID that you need.

Chapter 15. User Extensions Reference

To take advantage of the ISMF list capabilities, you can write your own REXX, CMS EXEC, EXEC 2 programs, or your own executable modules. These programs are known as user extension programs; they perform tasks against entries in a minidisk list, file list, management class list or a list of saved lists. For example, you may want to produce a printed report through a user extension program that extracts and formats information from a minidisk list. You initiate user extensions as commands from either the command line or the line operator field of list panels.

A sample user extension, PRNTMINI, is included on the DFSMS/VM Function Level 221 product tape. The sample user extension reads a saved minidisk list and creates a CMS output file that you can print. A listing of that program is in [Appendix G, “Sample User Extension Program — PRNTMINI,”](#) on page 241.

Initiating a User Extension

You can initiate a user extension either as a command or as a line operator in any application which supports lists. When you use an extension as a command, it affects all the entries in a list. When you use an extension as a line operator, it affects only the list entry against which you initiate it.

As a Command

To initiate a user extension as a command:

1. Display the list from which you want to initiate a user extension. It can be a minidisk list displayed on the Minidisk List panel, a list of files and directories displayed on the File List panel, a list of management class names displayed on the Management Class List panel, or a list of the saved lists displayed on the Saved ISMF Lists panel.
2. Enter EXECUTE followed by the name of your user extension, on the command line, followed by any parameters you want to pass.

You can also enter a slash (/) as a parameter. When "/" is used in this way, the actual parameters passed to the extension are decided by the application in which the extension is used.

- On the Minidisk List panel "/" is interpreted as the combination of minidisk owner ID and minidisk address, in that order. Only the first "/" is recognized.
- On the Management Class List panel, "/" is interpreted as the management class name. Only the first "/" is recognized as a substitution.
- On the Saved ISMF Lists panel, "/" is interpreted as the saved list name.
- On the File List panel, the interpretation depends on the character following the slash. A slash (/) followed by a blank is interpreted as the combination of file name, file type and file mode in that order. "/n" is the file name or directory name alone; "/t" is the file type alone; "/m" is the file mode alone; and "/d" is the fully qualified name of the parent directory. Combinations of these can also be used. For example, "/nt" is the file name and file type in that order; "/mtn" is the file mode, file type, and file name in that order; and so on. In the File Application, no characters other than these are supported following the slash.

[Figure 95 on page 172](#) shows a sample entry to use PRNTMINI as a command on the Saved ISMF Lists panel, with a slash (/) to indicate that you want to pass the list names as parameters.

ISMF displays a message, which is a summary message about the status of the requests issued, and places line operator history in the line operator column for each entry, as shown in [Figure 96 on page 172](#).

```

                                SAVED ISMF LISTS PANEL
COMMAND ==> EXECUTE PRNTMINI /
                                SCROLL ==> HALF
                                Entries 1-14 of 15
                                Data Columns 3-8 of 8
ENTER LINE OPERATORS BELOW:

  LINE   LIST   LIST   LIST   LIST   LIST   LIST ROW   LIST
  OPERATOR NAME  TYPE   DATE   TIME   USER  COUNT  UPDATE
  --- (1) --- -- (2) --- -- (3) --- -- (4) --- -- (5) --- -- (6) --- -- (7) --- -- (8) ---
          VLPROJ28 MINIDISK 93/01/04 09.16.25 USER01      14      3
          VLPROJ29 MINIDISK 93/01/04 10.07.18 USER01      51      3
          VLPROJ30 MINIDISK 93/01/04 11.05.39 USER01      73      1
          VLPROJ31 MINIDISK 93/01/05 08.27.31 USER01     106      1
          VLPROJ32 MINIDISK 93/01/05 09.17.58 USER01       8      2
          VLPROJ33 MINIDISK 93/01/05 09.47.34 USER01      16      1

```

Figure 95. Sample Entry to Initiate a User Extension as a Command

```

                                SAVED ISMF LISTS PANEL
COMMAND ==>
                                PRNTMINI SUCCESSFUL
                                SCROLL ==> HALF
                                Entries 1-14 of 15
                                Data Columns 3-8 of 8
ENTER LINE OPERATORS BELOW:

  LINE   LIST   LIST   LIST   LIST   LIST   LIST ROW   LIST
  OPERATOR NAME  TYPE   DATE   TIME   USER  COUNT  UPDATE
  --- (1) --- -- (2) --- -- (3) --- -- (4) --- -- (5) --- -- (6) --- -- (7) --- -- (8) ---
  *PRNTMINI VLPROJ28 MINIDISK 93/01/04 09.16.25 USER01      14      3
  *PRNTMINI VLPROJ29 MINIDISK 93/01/04 10.07.18 USER01      51      3
  *PRNTMINI VLPROJ30 MINIDISK 93/01/04 11.05.39 USER01      73      1
  *PRNTMINI VLPROJ31 MINIDISK 93/01/05 08.27.31 USER01     106      1
  *PRNTMINI VLPROJ32 MINIDISK 93/01/05 09.17.58 USER01       8      2
  *PRNTMINI VLPROJ33 MINIDISK 93/01/05 09.47.34 USER01      16      1

```

Figure 96. Sample Line Operator History for a User Extension

As a Line Operator

To invoke a user extension program as a line operator:

1. Display the list from which you want to initiate a user extension. It can be a minidisk list displayed on the Minidisk List panel, a list of files and directories displayed on the File List panel, a list of management class names displayed on the Management Class List panel, or a list of the saved lists displayed on the Saved ISMF Lists panel.
2. Enter the name of your user extension, followed by any parameters you want to pass, in the line operator field next to the entry against which you want to initiate the extension. If needed, you can type over data in other fields of the list entry.

You can also use a slash as a parameter. The usage is described in the preceding section.

Figure 97 on page 172 shows a sample entry.

```

                                SAVED ISMF LISTS PANEL
COMMAND ==>
                                SCROLL ==> HALF
                                Entries 1-14 of 15
                                Data Columns 3-8 of 8
ENTER LINE OPERATORS BELOW:

  LINE   LIST   LIST   LIST   LIST   LIST   LIST ROW   LIST
  OPERATOR NAME  TYPE   DATE   TIME   USER  COUNT  UPDATE
  --- (1) --- -- (2) --- -- (3) --- -- (4) --- -- (5) --- -- (6) --- -- (7) --- -- (8) ---
  PRNTMINI / VLPROJ28 MINIDISK 93/01/04 09.16.25 USER01      14      3
              VLPROJ29 MINIDISK 93/01/04 10.07.18 USER01      51      3
              VLPROJ30 MINIDISK 93/01/04 11.05.39 USER01      73      1
              VLPROJ31 MINIDISK 93/01/05 08.27.31 USER01     106      1
              VLPROJ32 MINIDISK 93/01/05 09.17.58 USER01       8      2
              VLPROJ33 MINIDISK 93/01/05 09.47.34 USER01      16      1

```

Figure 97. Sample Entry to Initiate a User Extension as a Line Operator

You can repeat a user extension line operator by entering an equal sign (=) next to any subsequent entries as shown in Figure 98 on page 173.

ISMF displays a short message that indicates the status of the last request issued and places line operator history in the line operator column for each entry. An example is shown in Figure 99 on page 173.

```

COMMAND ==>>
                                SAVED ISMF LISTS PANEL
                                SCROLL ==>> HALF
                                Entries 1-14 of 15
                                Data Columns 3-8 of 8
ENTER LINE OPERATORS BELOW:

  LINE   LIST   LIST   LIST   LIST   LIST   LIST ROW   LIST
OPERATOR NAME  TYPE   DATE   TIME   USER  COUNT  UPDATE
---(1)--- --(2)--- --(3)--- --(4)--- --(5)--- --(6)--- --(7)--- --(8)---
PRNTMINI / VLPROJ28 MINIDISK 93/01/04 09.16.25 USER01      14      3
=          VLPROJ29 MINIDISK 93/01/04 10.07.18 USER01      51      3
          VLPROJ30 MINIDISK 93/01/04 11.05.39 USER01      73      1
=          VLPROJ31 MINIDISK 93/01/05 08.27.31 USER01     106      1
          VLPROJ32 MINIDISK 93/01/05 09.17.58 USER01       8       2
=          VLPROJ33 MINIDISK 93/01/05 09.47.34 USER01      16      1

```

Figure 98. Repeating a User Extension Entered as a Line Operator

```

COMMAND ==>>
                                SAVED ISMF LISTS PANEL
                                SCROLL ==>> HALF
                                Entries 1-14 of 15
                                Data Columns 3-8 of 8
ENTER LINE OPERATORS BELOW:

  LINE   LIST   LIST   LIST   LIST   LIST   LIST ROW   LIST
OPERATOR NAME  TYPE   DATE   TIME   USER  COUNT  UPDATE
---(1)--- --(2)--- --(3)--- --(4)--- --(5)--- --(6)--- --(7)--- --(8)---
*PRNTMINI VLPROJ28 MINIDISK 93/01/04 09.16.25 USER01      14      3
*PRNTMINI VLPROJ29 MINIDISK 93/01/04 10.07.18 USER01      51      3
          VLPROJ30 MINIDISK 93/01/04 11.05.39 USER01      73      1
*PRNTMINI VLPROJ31 MINIDISK 93/01/05 08.27.31 USER01     106      1
          VLPROJ32 MINIDISK 93/01/05 09.17.58 USER01       8       2
*PRNTMINI VLPROJ33 MINIDISK 93/01/05 09.47.34 USER01      16      1

```

Figure 99. Saved ISMF Lists Panel After Completion of the Line Operator

Accessing a Saved List through EXECs

This section is intended to help you use an EXEC to access information in a saved list, as an example of operations that can be performed on an ISMF list.

You can write your own EXECs to access and extract information from saved lists. Saved lists are stored as ISPF tables. Use ISPF table services, such as TBOPEN, TBGET, and TBCLOSE to access those lists, refer to the *ISPF for VM Dialog Management Guide*. In your EXECs, use the variable names for data columns in the lists to retrieve the specific information you need. See [Appendix F, “ISMF Variables,”](#) on page 233 for lists of the ISMF variable names.

You can use EXECs to access a saved list in one of two ways:

- As a user extension invoked as a command or line operator on the Saved ISMF Lists panel in the ISMF environment
- As an EXEC invoked outside of ISMF in an ISPF environment

Initiating an EXEC as a Line Operator

When you use an EXEC as a user extension from the Saved ISMF Lists panel, ISMF can automatically get the name of the saved list you specify and define it to ISPF. In your user extension, you can acquire the name of the saved list against which you have initiated your user extension. The list name is in the data column with variable name COBJ. You can specify a REXX statement such as:

```
ADDRESS ISPEXEC 'VGET COBJ SHARED';
```

After obtaining the list name, it can be used to open the saved list.

```
ADDRESS ISPEXEC 'TBOPEN' COBJ;
```

Initiating an EXEC Outside of ISMF

To use an EXEC to access saved lists outside of ISMF, you must define to ISPF the macro libraries in which your saved lists are stored. Use a FILEDEF statement specifying the DGTTLIB reserved ddname known to ISPF.

For example, you could specify:

```
'FILEDEF DGTTLIB DISK fn MACLIB fm (CONCAT '
```

where:

DGTTLIB

is a reserved ddname known to ISPF.

fn

is the file name of the macro library where your saved list is stored.

MACLIB

is the file type of your saved list.

fm

is the file mode of the macro library where your saved list is stored.

CONCAT

specifies that you want to concatenate this FILEDEF statement with other FILEDEF statements by the same name.

Alternately, you can use the LIBDEF service with a FILEDEF statement to define the macro libraries in which your saved lists are stored to ISPF, using a ddname of your choice (for additional information about FILEDEF and LIBDEF, refer to the *ISPF Dialog Management Guide*):

- Issue a FILEDEF statement to associate your saved list with a ddname of your choice:

For example, you could specify:

```
'FILEDEF myddname DISK fn MACLIB fm (CONCAT '
```

where:

myddname

is a name of your choice.

fn

is the file name of the macro library where your saved list is stored.

MACLIB

is the file type of the macro library where your saved list is stored.

fm

is the file mode of the macro library where your saved list is stored.

CONCAT

specifies that you want to concatenate this FILEDEF statement with other FILEDEF statements by the same name.

- Issue a LIBDEF statement to associate the ddname you have chosen with the DGTTLIB reserved ddname known to ISPF.

For example, you could specify:

```
ADDRESS ISPEXEC 'LIBDEF DGTTLIB LIBRARY ID(myddname)'
```

where:

DGTTLIB

is a reserved ddname known to ISPF.

myddname

is a name of your choice defined by the FILEDEF statement.

You also need to use the saved list variable names in your EXEC to retrieve specific information from the saved list you specify. See [Appendix F, “ISMF Variables,”](#) on [page 233](#) for the variable names, attributes, and lengths for a saved list.

[Figure 100](#) on [page 175](#) is a code segment that performs the retrieval of information from the first row of the saved minidisk list SAMPLE from DGTTABL MACLIB. This code can be invoked like any other ISPF application. It can also be invoked from the ISMF command line.

```
/* get first row */
'filedef dummy disk dgttabl maclib * ( concat'
ADDRESS ISPEXEC
'LIBDEF ISPTLIB LIBRARY ID(DUMMY)'
'TBOPEN SAMPLE NOWRITE SHARE'
if rc = 0 then Do
'TBSKIP SAMPLE'
if rc = 0 then Do
'TBGET SAMPLE '
if rc = 0 then Do
say 'First row retrieved from SAMPLE minidisk list, values are:'
say 'Display Flag          = ' ydispflg
say 'Filter  Flag          = ' yfiltflg
say 'Hide   Flag          = ' yhideflg
say 'Line Operator         = ' ylineop
say 'Minidisk list entry   = ' ybjec
say 'Minidisk start        = ' ydstart
say 'Minidisk end          = ' ydend
say 'Minidisk size         = ' ydsize
say 'Minidisk gap          = ' ydgap
say 'Device VOLSER         = ' yevvolser
say 'Device type           = ' yevtype
say 'Duplex status         = ' yplxstat
say 'Fast write status     = ' ydfwstat
say 'Read cache status     = ' ydcastat
say 'Cache fast write status = ' yafwstat
say 'System ID             = ' ysystemid
say 'Minidisk address      = ' ydbaddr
say 'Minidisk start        = ' ydbstart
say 'Minidisk end          = ' ydbend
say 'Request ID            = ' yequstid
say 'Short message         = ' ysmg
say 'Long message          = ' ylmsg
say 'Minidisk owner        = ' yobj
say 'Minidisk address      = ' ydaddr
end
'TBCLOSE SAMPLE'
```

Figure 100. Sample Code Segment

Chapter 16. ACS Language Reference

PI

This chapter contains Programming Interface Information.

Automatic class selection (ACS) routines are for automatically assigning management classes to files and directories. The ACS routines are written in the ACS language, a high-level programming language. After writing the routines, you must translate them into an object form that DFSMS/VM understands (see [“Translate ACS Routines” on page 67](#)). A successful translation places the ACS object in a specified source configuration file. After you activate the configuration contained in that source configuration file, the ACS routine governs storage management.

This chapter contains four topics. The first section describes the ACS language constants. The second section describes *read/write variables*. The third section describes *read-only variables* that the ACS routine uses for comparison operations. The fourth section describes the ACS language statements. For samples of the ACS language statements used in ACS routines, see [Appendix D, “Sample ACS Routines,” on page 225](#).

ACS Language Constants

You can use four types of constants in ACS routines:

Numeric

A numeric constant is a string consisting of up to ten characters, 0 through 9. You can use numeric constants in comparison operations involving numeric variables, for example, &NDIRS and &SIZE.

KB, MB

KB and MB are suffixes for numeric constants, such as 200KB and 10MB. You can only use KB and MB in comparison operations involving the &SIZE read-only variable. The maximum prefix value for KB is 21 4748 3647. KB is the abbreviation for a kilobyte, which has the value of 1024 bytes. The maximum prefix value for MB is 209 7151. MB is the abbreviation for a megabyte which has the value of 104 8576 bytes.

Literal

A literal is a character string, such as 'VMSYS1.DFSMS22', that is enclosed in single quotation marks. The maximum length of a literal is 255 characters. If you want a literal to contain a single quotation mark, such as PAYROLL'SDATA, you must specify two single quotation marks: 'PAYROLL'SDATA'. All literal values are shifted to uppercase. The literal " indicates NULL, which can be used with various ACS variables.

Mask

A mask is a character string, such as VMSYS1.*22, that is not enclosed in single quotation marks. You can use a mask to represent a file name, file pool name, or other values that have a common string of characters.

A mask must begin with an alphabetic character, a numeric character (0–9), a national character (\$, @, #), or an underscore. The asterisk (*) and percent sign (%) have special significance in a mask. The following sections describe the rules for using these characters in simple masks.

Simple Mask Rules

The following rules apply to the special characters in a simple mask.

- An ***** matches zero or more characters.
- Two or more adjacent asterisks are not allowed within a simple mask.
- A **%** is a single character place holder. **%%%** represents three character positions.

Simple Mask Examples

ABC*

All strings of any length beginning with ABC

XYZ

All strings of any length having three adjacent characters XYZ

DIR%%%

All six-character strings beginning with DIR

***%WK%%**

All strings where the second and third characters of the last five (or only five) are WK

Directory Path Mask Rules

The following rules apply to the special characters in a directory path mask:

- Separate directory path levels with periods, (.). There is a maximum of eight levels.
- The first directory path level (user ID) has a maximum length of eight characters. The remainder of the levels each have a maximum length of sixteen characters. The maximum length of the entire path is 144 characters.
- A **%** is a single character place holder. “%%%” represents three character positions.
- A ***** within a directory path level means that zero or more characters are represented.
- A single ***** can also represent a single directory path level place holder.
- A ****** means that zero or more directory path levels are represented.
- A ****** cannot appear with any other characters within a directory path level.
- Three or more adjacent ***** are not allowed within a directory path level.

Directory Path Mask Examples

****OUTLIST**

All names where the last directory path level is “OUTLIST”. This example means that zero or more directory path levels are represented.

.PAYROLL.*SALARY.

All names with six directory path levels where the third directory path level is “PAYROLL” and the fifth directory path level is “SALARY”.

***.%%TEST.*.DATA**

All names with four directory path levels where the second directory path level has six characters ending in “TEST” and the fourth directory path level is “DATA”.

****.*ABC*.***

All names where some (or only) directory path level contains the characters “ABC”.

LABMGR..DATA**

All names where the first directory path level is “LABMGR” and the last directory path level is “DATA”.

ACS Language Read/Write Variables

Use ACS routines to assign values to read/write (R/W) variables. Of the two types of variables, read/write and read-only, only the read/write variables can be modified. In DFSMS/VM, the only variable which can be modified is **&MGMTCLAS**. This variable can have a value assigned to it by the SET statement and it can also be used in comparison operations. It has an initial value that can be altered by the ACS routine.

During ACS processing, the &MGMTCLAS variable is set by DFSMS/VM and passed to the ACS routine at invocation time. The initial value that is available as input to the ACS routine varies with the type of ACS processing that invokes the ACS routine. The initial value of the &MGMTCLAS variable for:

- File or directory creation, is NULL

- File recall, is:
 - The file's current management class name
 - NULL, if the file has the NULL management class
 - NULL, if the file has no management class
- Conversion processing, is:
 - The file's current management class name, if it has a management class and the DFSMS CONVERT command is issued with the REDETERMINE option
 - NULL, if the file or directory has the NULL management class and the DFSMS CONVERT command is issued with the REDETERMINE option
 - NULL, if the file or directory has no management class

In an ACS routine, the following values can be assigned to the &MGMTCLAS variable:

- A alphanumeric constant enclosed in single quotes. This value needs to be a valid management class name, see “Management Class Name” on page 39. If this value is not valid the ACS routine can be translated, but the configuration containing the ACS routine is invalid.
- The variable &MGMTCLAS.
- A NULL value by specifying two single quotes ("").
- The variable &PARENTMC.
- The variable &DEF_MGMTCLAS. For example, the NULL value is assigned to &MGMTCLAS, since the read only variable &DEF_MGMTCLAS is initialized to NULL.

Note: The NULL management class is generated when the NULL value is assigned (explicitly or implicitly) to the &MGMTCLAS variable.

The following variables are initialized to NULL:

- &STORCLAS
- &DATACLAS
- &STORGRP

ACS Language Read-Only Variables

The majority of the ACS variables are read-only. Read-only variables contain file and directory information, and they reflect what is known at the time of the request. You can use read-only variables in comparison operations, but you cannot change their values.

The values of the read-only variables are set by DFSMS/VM and passed to the ACS routine at invocation time. The values are based on what is known at the time ACS is invoked. For example, the value of **&SIZE** is set to zero when a new file is created, but on a recall or convert request, it is set to the actual size of the file.

The following lists all of the DFSMS/VM read-only variables, and the following pages explain the contents of the read-only variables.

&ACSENVIR
 &DIR
 &FN
 &FT
 &FPOOLID
 &NDIRS
 &OWNER
 &PARENTMC
 &PATH
 &RECFM

&SIZE
&USER

NAME	DESCRIPTION
------	-------------

&ACSENVIR	
----------------------	--

The environment from which the ACS routine was invoked, one of:

CREATE	
---------------	--

for creation of files or directories

RECALL	
---------------	--

for file recall operations

CONVERT	
----------------	--

for file or directory convert operations initiated by the DFSMS CONVERT, DFSMS MANAGE, or DFSMS MIGRATE commands

other	
--------------	--

if it is blank or binary zeros, it is interpreted as a CREATE: otherwise, it allows the ACS module exit routine to set its own value before reinvoking the ACS routine

```
Type: Literal
Max length: 8 characters
```

&DIR	
-----------------	--

The name of the directory being created for which ACS processing is taking place. This variable is NULL when a file is processed.

```
Type: Literal
Max length: 16 characters
```

&FN	
----------------	--

The name of the SFS file for which ACS processing is taking place. This variable is NULL when a directory is processed.

```
Type: Literal
Max length: 8 characters
```

&FT	
----------------	--

The type of the SFS file for which ACS processing is taking place. This variable is NULL when a directory is processed, for FN and FT.

```
Type: Literal
Max length: 8 characters
```

&FPOOLID	
---------------------	--

The name of the file pool in which the directory or the file for which ACS processing is taking place resides.

```
Type: Literal
Max length: 8 characters
```

&NDIRS	
-------------------	--

The number of directory levels in **&PATH**. When ACS processing is performed on a top-level directory, this variable is initialized to *zero*.

```
Type: Numeric
Max value: 9
```

&OWNER	
-------------------	--

The owner of the directory or base file for which ACS processing is taking place.

```
Type: Literal
Max length: 8 characters
```

&PARENTMC

The management class of the directory specified in **&PATH**. This variable is NULL when the top directory is processed, or the directory has the null management class, or the directory has no management class.

```
Type: Literal
Max length: 8 characters
```

&PATH

The directory name, excluding the file pool ID, derived from the full name of the directory in which the SFS base file or directory to be processed is located. When ACS processing is performed on a top directory, **&PATH** is initialized with the NULL string. This variable is a string of characters defined as follows:

```
userid.d1.d2.d3 ... .d8
```

Where:

d1.d2. ... d8

are the subdirectory qualifiers. Subdirectory names are up to 16 characters in length. The first character must be alphabetic, but the remainder can be alphabetic or numeric. The period between *userid* and any further directories is required.

```
Type: Literal
Max length: 144 characters
```

Note: For directory processing, the path does not include the directory being processed. For example, if directory VMSYSU:TEST.DIRECTORY.ONE is being created, the **&PATH** variable contains TEST.DIRECTORY and **&DIR** contains ONE. For file processing, recalling TEST FILE VMSYSU:TEST.DIRECTORY.ONE **&PATH** would contain TEST.DIRECTORY.ONE.

&RECFM

The record format of the file for which the ACS is taking place. Valid values are **F** (fixed format records) and **V** (variable format records). This variable is NULL when a directory is processed.

```
Type: Literal
Values: Either F or V
```

&SIZE

Size (in KB or MB) of the file to be processed, it defaults to zero if the size is unknown.

```
Type: Numeric, abutted with either KB or MB
Max value: 2147483647 for KB; 2097151 for MB
```

&USER

The user ID of the person taking action that leads to ACS invocation. This would be the person creating, recalling, or converting the file or directory.

```
Type: Literal
Max length: 8 characters
```

The following lists the read-only variables (not supported by DFSMS/VM) that are available in MVS but are initialized to zero by DFSMS/VM.

&MAXSIZE
&MEMNQUAL
&NQUAL
&NVOL
&RETPD

The following lists the read-only variables (not supported by DFSMS/VM) that are available in MVS but are initialized to NULL.

&ACCT_JOB
&ACCT_STEP
&ALLVOL
&ANYVOL
&APPLIC
&DD
&DEF_DATACLAS
&DEF_MGMTCLAS
&DEF_STORCLAS
&DSN
&DSNTYPE
&DSORG
&DSOWNER
&DSTYPE
&EXPDT
&GROUP
&HLQ
&JOB
&LLQ
&MEMHLQ
&MEMLLQ
&MEMN
&MSVGP
&PGM
&RECORG
&UNIT
&XMODE

Comparison Operators

Comparison operators allow you to determine the difference between two values. The following comparison operators are allowed:

This:

Means this:

GT or >

Greater than

LT or <

Less than

NG or ¬>

Not greater than

NL or ¬<

Not less than

EQ or =

Equal

NE or ¬=

Not equal

GE or >=

Greater than or equal

LE or <=

Less than or equal

When performing comparisons alphabetic characters compare lower than digits (A–Z come before 0–9). The following comparison is true for all file names alphanumerically greater than M:

```
IF &FN > 'M' THEN . . .
```

For FILTLIST or mask comparisons, only EQ and NE are valid. See [“FILTLIST” on page 185](#) for details.

Comparison Rules

The following rules apply to comparisons:

1. For a valid comparison, one operand must either be a read-only variable or a read/write variable and the other operand must be a constant (any of the four types), a read-only variable, or a FILTLIST name.
2. Numerics are right justified.
3. Literals are left justified and padded with blanks.
4. Type checking is done to ensure that numeric read-only variables are not being compared to characters (literals) and that character (literal) read-only variables are not being compared to numbers.
5. Limited length checking of read-only variables is performed to ensure that the maximum lengths are not exceeded. For example, a literal to which &PATH is being compared must be no longer than 144 characters.

See [“ACS Language Read-Only Variables” on page 179](#) for maximum lengths.

Boolean Expressions

You can use the following Boolean operators in any ACS routine:

This:

Means this:

AND or &&

And

OR or |

Or

AND takes precedence over OR. In the following example, condition 1 and condition 2, together with the AND operator, are evaluated first; their result is then evaluated with condition 3 and the OR operator.

```
IF (condition 1) AND  
   (condition 2) OR  
   (condition 3) THEN
```

You can group relational expressions by using sets of parentheses. You can combine these groupings with the Boolean operators to form more complicated relational expressions. In the following example, each set of the expressions within parentheses is evaluated first; the two results are then evaluated.

```
IF (condition 1 AND condition 2) AND  
   (condition 3 OR condition 4) THEN . . .
```

ACS Directory-Level Indexing Function

This function allows the storage administrator to index the **&PATH** variable (for accessing a particular directory):

- **&PATH(1)** — first directory of the path name (*userid*), is the top directory.
- **&PATH(2)** — second directory of the directory path name. For example d1 from *userid.d1*.
- **&PATH(&NDIRS)** — last directory of the directory path name.

The valid values for indexes are:

- Integers (1–9)
- The read-only variable **&NDIRS**

ACS Language Statements

This section describes the function and syntax of the ACS language statements that you can use for writing ACS routines.

The continuation characters '+' (plus sign) and '-' (minus sign) allow you to extend literal constants to the next line. To ignore the leading blanks on the following line, use '+'. If you want to include the leading blanks on the next line as part of a literal, then use a '-'. You cannot continue masks, numbers, KB or MB numerics, or keywords. All input statements are shifted to uppercase characters.

The maximum number of nesting levels for any combination of ACS statement types is 32. (For example, a nested IF statement is one that appears within an IF statement.)

Comments begin with a slash-asterisk pair, '/*', and end with an asterisk-slash pair, '*/'. You can place comments anywhere within an ACS routine except on a line containing a continuation character.

The statement types are defined as follows:

Statement	Description
-----------	-------------

PROC	Start of an ACS routine.
-------------	--------------------------

FILTLIST	Definition of filter criteria.
-----------------	--------------------------------

SET	Assigns a value to a read/write variable.
------------	---

DO	Start of statement group.
-----------	---------------------------

IF	Provides conditional statement execution.
-----------	---

SELECT	Defines a set of conditional execution statements.
---------------	--

EXIT	Causes immediate ending of the ACS routine and can be used to fail processing of a file or directory.
-------------	---

WRITE	Writes messages to the DFSMS/VM log. These messages are informational messages and are logged if the installation has specified this level for logging.
--------------	---

END	End of statement group (for example, DO, SELECT) or ACS routine (PROC).
------------	---

PROC

PROC is the first statement of each ACS routine. It identifies the ACS routine and the read/write variable the routine sets. You can precede the PROC statement with blank lines or comments, but not with other statements. You can place a blank and then a number, such as 0 or 1, after the PROC statement. The number does not affect ACS language processing. To identify an ACS routine and the value it is to determine, you must specify a read/write variable at the end of the PROC statement. You must also place an END statement at the end of each ACS routine.

The symbols < and > denote that the contents between the symbols are optional.

Syntax

PROC <*n*> *read/write variable* :: **END**

- *n* is optional and can contain any numeric value.
- *read/write variable* must be MGMTCLAS or &MGMTCLAS.

FILTERLIST

The FILTERLIST statement is a definition list that you can use for testing variables in an ACS routine. You define the information that you want to include and exclude in the list by using the INCLUDE and EXCLUDE keywords. Then you can compare read-only variables to items in the list by using IF-THEN and SELECT-WHEN statements, without having to write elaborate AND and OR combinations.

FILTERLIST is a definition statement that simplifies comparison operations. It is not a statement that initiates processing, and it does not change the value of any variables.

Because a FILTERLIST can contain only literal values, you can only compare it to literal read-only variables. This excludes the numeric read-only variables from FILTERLIST comparisons.

You must define a FILTERLIST before you refer to it in the body of an ACS routine.

Syntax

FILTERLIST *name* <INCLUDE(*list*)> <EXCLUDE(*list*)>

- *name* is mandatory and can be at most 32 alphanumeric characters in length. You can also use an underscore, `_`. In the FILTERLIST, you can optionally precede the name with an ampersand, `&`. If you omit the ampersand it is added during translation and is always counted as part of the FILTERLIST name. When referring to the FILTERLIST in the body of the routine, you must always precede the FILTERLIST name with an ampersand.
- You must specify INCLUDE, EXCLUDE, or both in the FILTERLIST statement. If a list item satisfies both the INCLUDE and EXCLUDE criteria, EXCLUDE takes precedence and prevents the item from being included in the list.
- *list* can contain literals, simple masks, and directory path masks.
- You can specify as many as 255 list items in each INCLUDE(*list*) and EXCLUDE(*list*).

Example:

```
PROC MGMTCLAS  
  
  FILTERLIST VLIST2 INCLUDE(DBX*, LAB*) EXCLUDE('DBX191', 'LAB256')  
  
  IF &USER = &VLIST2 THEN  
    (some action)  
  
END
```

In the environment shown in [Figure 101 on page 186](#), the value of the IF statement is true for any of the following user IDs:

LAB191
LAB002
DBX256
DBXRES

The value of the IF statement is false for the following user IDs, because they match the EXCLUDE filter criteria:

LAB256
DBX191

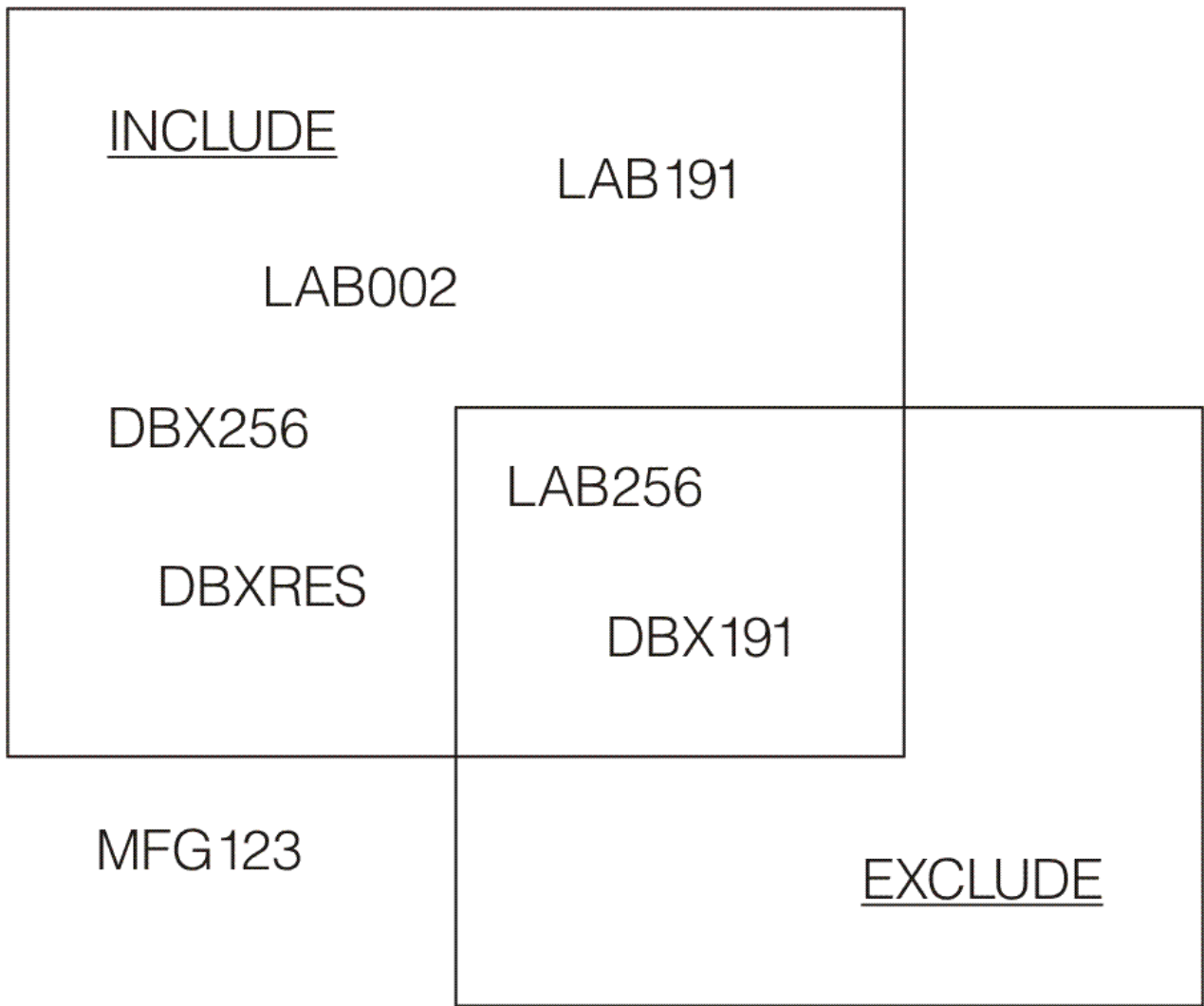


Figure 101. Using INCLUDE and EXCLUDE

SET

The SET statement is used to assign a value to the &MGMTCLAS variable. This value is either a management class name or NULL. The SET statement supports the assignment of &PARENTMC or &DEF_MGMTCLAS to &MGMTCLAS.

The name is 1–8 characters long, and must be enclosed in single quotation marks. If a management class name is assigned, the name must be a valid management class name (see [“Management Class Name”](#) on page 39).

You cannot set &MGMTCLAS equal to a FILTLIST variable or a read-only variable. For example, the following statement is invalid.

```
SET &MGMTCLAS = &FN
```

If the statement is used, you receive the following error message.

```
IGDST3224I Invalid read/only variable &FN on SET statement
```

You can assign a NULL value to the read/write variable by specifying two single quotation marks with nothing between them (").

Syntax

SET &MGMTCLAS = *value*

- You can specify EQ in place of the equal sign (=).
- *value* can be one name or a NULL value.

Examples.

```
SET &MGMTCLAS = 'normal'
SET &MGMTCLAS EQ 'normal'
SET &MGMTCLAS = ''
SET &MGMTCLAS EQ &PARENTMC
```

DO

You can group a collection of ACS language statements by using a DO statement paired with an END statement. The DO statement follows an IF-THEN clause, an ELSE clause, or a SELECT-WHEN group.

Syntax

DO *group of statements* **END**

A *group of statements* can consist of zero or more ACS language statements.

Example:

```
IF &FN = 'PAYROLL' THEN
DO
WRITE 'No Payroll allowed'
EXIT CODE(1)
END
```

IF

Use the IF statement for conditional statement execution. You must always follow an IF statement with a THEN clause. The THEN clause can be a single statement or a DO-END group of statements. You can also follow a THEN clause with an ELSE clause. If you specify the ELSE clause, you must follow it with either a single statement or a DO-END group of statements. If you want to specify the ELSE clause, but do not want to follow it with an executable statement, follow it with an empty DO-END pair.

If the result of the IF statement comparison is true, the THEN clause is executed. If the result is false, the ELSE clause is executed. If you omit the ELSE, processing continues with the next sequential statement after the THEN clause.

Syntax

IF *relational expression* THEN *clause* <ELSE *clause*>

- *relational expression* is either a single comparison or multiple comparisons joined by Boolean operators.
- THEN is a mandatory keyword.
- *clause* contains a single statement, a DO-END group of statements, or a SELECT statement.
- ELSE is an optional keyword.
- An ELSE *clause* is mandatory if you specify the ELSE keyword.

Note: For information on what constitutes a valid relational expression, see [“Comparison Rules” on page 183](#).

Example:

Example of a specified ELSE:

Example of a NULL ELSE:

<pre> IF &USER = 'BILL' THEN SET &MGMTCLAS = 'ONE' ELSE SET &MGMTCLAS = 'TWO' </pre>	<pre> IF &USER = 'BILL' THEN SET &MGMTCLAS = 'ONE' ELSE DO END SET &MGMTCLAS = 'TWO' </pre>
--	---

The statement on the left sets &MGMTCLAS equal to ONE or TWO, depending on the value of &USER.

The statement on the right sets &MGMTCLAS equal to TWO, regardless of the value of &USER. If &USER equals 'BILL', then &MGMTCLAS is set to 'ONE'. The ELSE clause is skipped, and execution falls to the next statement, which changes &MGMTCLAS to 'TWO'. If &USER does not equal 'BILL', execution falls to the ELSE, which results in no assignment to &MGMTCLAS. Then execution proceeds to the next sequential statement, which sets &MGMTCLAS to 'TWO'.

SELECT

Use the SELECT statement to write conditional statements in sequential form rather than IF-THEN-ELSE form. A SELECT statement consists of a SELECT keyword, one or more WHEN clauses, an optional OTHERWISE clause, and an END statement. You can specify the SELECT statement in one of two forms. In the first form, you include the variable being tested after the SELECT statement. In the second form, you include the variable being tested after the WHEN keyword.

In either form, the first true WHEN condition is executed, and the remaining WHEN conditions are ignored.

Syntax

First Form

Implicit equality comparison between the variable and the value.

SELECT (*variable*) WHEN (*value*) <action> . . . <WHEN (*value*) <action>> <WHEN (*value*) <action>> <OTHERWISE <action>> END

- *variable* is any read-only or read/write variable, except &SIZE.
- At least one WHEN keyword is mandatory.
- *value* is a constant or a FILTLIST name.
- *action* is a valid ACS statement, SELECT group, or a DO END group, but cannot be blank.
- OTHERWISE is an optional keyword.

Example:

```

SELECT (&USER)
  WHEN ('IBMUSER1') SET &MGMTCLAS = 'PAYROLL'
  WHEN ('IBMUSER2') SET &MGMTCLAS = 'TEST'
  WHEN ('IBMUSER3') SET &MGMTCLAS = 'DEVELOP'
  OTHERWISE        SET &MGMTCLAS = 'NORMAL'
END

```

Second Form

Explicit variable - value comparison

SELECT WHEN (*relational expression*) <action> . . .

<WHEN (*relational expression*) <action>>

<WHEN (*relational expression*) <action>>

<OTHERWISE <action>> END

- At least one WHEN keyword is mandatory.
- *relational expression* is either a single comparison or multiple comparisons joined by Boolean operators.

- The read-only variable &SIZE is allowed in a relational expression for this form of the SELECT statement.
- *action* is a valid ACS statement, SELECT group, or a DO END group, but cannot be blank.
- OTHERWISE is an optional keyword.

Example

```
SELECT
  WHEN (&USER = 'IBMUSER') SET &MGMTCLAS = 'PAYROLL'
  WHEN (&USER = 'IBMUSER2')

    IF &OWNER = 'UT34' THEN
      SET &MGMTCLAS = 'TEST'
    ELSE
      SET &MGMTCLAS = 'EVERYONE'

  WHEN (&FT = 'TEMP') SET &MGMTCLAS = ''
  WHEN (&FN = 'CADAM') SET &MGMTCLAS = ''
  OTHERWISE SET &MGMTCLAS = 'COMMON'
END
```

EXIT

The EXIT statement immediately terminates the operation of an ACS routine.

Syntax

EXIT <CODE(*n*)>

- CODE(*n*) is an optional keyword, but you must specify *n*, an exit code value. A nonzero value for *n* causes the subsequent ACS processing to be skipped and the rejection of ACS processing. The rejection causes a create or recall to fail and a convert does not make any changes to the management class of the file or directory being processed.

EXIT Example

```
PROC MGMTCLAS
  FILTLIST NOEXP INCLUDE (*) /*Classes that don't expire */

  IF (&ACSENVIR = 'CREATE' | &ACSENVIR = 'CONVERT')
    THEN
      DO
        SET &MGMTCLAS = 'EX7DMI00'
        EXIT
      END
  IF &PARENTMC = &NOEXP THEN EXIT CODE(22)
  :
END
```

If the first IF statement is *true* (&ACSENVIR is equal to CREATE or CONVERT), then set &MGMTCLAS and exit.

If the environment is not create or convert, then processing continues until the ACS routine encounters the next statement where &PARENTMC is checked to see whether a file should be expired. If it does not expire the file, ACS processing stops and rejects the processing of the file.

WRITE

Use the WRITE statement to log a message while processing the ACS routine. The message can be either a string or a combination of variables and strings. The maximum length of the message, after substitution of values for any variable is 250 characters. All numeric values are displayed in hexadecimal.

You must enclose all text strings in single quotation marks. If you want a single quotation mark to be part of the message, use two single quotation marks to represent it.

This:

Appears as this:

WRITE 'This line's short.'

This line's short.

You can use continuation characters (+, -) to continue text onto a subsequent line. The closing single quotation mark signifies the end of text.

A header is created with the message number and the system information. The informational message is only saved if logging of informational messages has been requested in the control file.

Syntax

WRITE 'message'

- *message* is passed to the storage administrator through the DFSMS log.

Example:

Assuming the value of &MGMTCLAS is 'MC1', (and informational message logging has been requested) the following WRITE message:

```
WRITE 'WARNING - &MGMTCLAS SPECIFIED (' &MGMTCLAS ';')&MGMTCLAS IS NOT ALLOWED'
```

Displays the header and the message as follows:

```
IGDSZ01009I WARNING - &MGMTCLAS SPECIFIED (MC1) IS NOT ALLOWED
```

A maximum of five messages can be logged for each file or directory processed. If any more messages are generated, a sixth and final message indicates that additional messages have been generated, but the additional messages are not displayed.

END

The END statement concludes an ACS routine, a DO group, or a SELECT statement.

Syntax

END

Example:

```
PROC MGMTCLAS
:
: (source code)
:
: END
```

Appendix A. ACS REXX Exit

PI

This appendix contains Programming Interface Information.

A sample ACS REXX exit (SMPREXX EXEC) and a sample configuration (SMPCNFGR CONFIG) are provided in the sample disk on the DFSMS/VM product tape.

General Information

DFSMS/VM allows an installation-written REXX exit to take control after the ACS routine finishes processing and before the module exit starts processing. Alternatively, the ACS REXX exit can be used instead of the ACS routine or ACS module exit to do the following:

- Modify management classes.
- Log customized informational messages using the DFSMS subcommand WRITE. If you want informational messages displayed on the console or logged to the DFSMS log file, set the values for DFSMS_LOG_TO_CONSOLE and DFSMS_LOG_TO_FILE in the DFSMS control file to 4.
- Reject ACS processing. If the ACS REXX exit rejects ACS processing, the file is not migrated or recalled, or the file or directory is not created. If a DFSMS CONVERT or MANAGE command has been issued, no changes are made to the management class of the file or directory currently processing.

Note: If the ACS REXX exit is used, IBM recommends the use of compiled REXX.

DFSMS/VM provides subcommands that enable the REXX exit to perform the same functions as the ACS routine. The REXX exit can perform comparisons using the ACS read-only and read/write variables and sets the management class assigned to the file or directory for which ACS processing is being performed. The REXX exit can also write messages to the DFSMS/VM log.

The REXX exit is also capable of performing tasks not achievable from the ACS routine. For example, it performs arithmetic operations, issues CMS or CP commands, and invokes external routines.

Every effort must be made to avoid infinite loops, invocation of external routines that can abend, assignment of invalid management classes, or other situations that degrade the performance or reliability of the system. For example, invoking ACS on a file create with a REXX EXEC in a file pool being managed by DFSMS is a scenario that causes an infinite loop.

Using the IBM Compiler for REXX/370

You can run ACS REXX exits using the VM/REXX Interpreter; however, improved performance is achieved through the REXX/370 compiler. The compiler also performs syntax checking of the entire REXX exit, thereby eliminating the possibility of syntax errors during run time. REXX/370 consists of two components, the IBM Compiler for REXX/370 (program number 5695-013) and the IBM Library for REXX/370 (program number 5695-014). Because run-time support is required for execution of compiled REXX programs, this support must be installed on the same system as DFSMS/VM if compiled REXX exits are used (that is, the DFSMS master and space management server machines must have access to the disk that contains the run-time library). The REXX/370 library is required when running compiled REXX programs.

Once the REXX/370 compiler is installed, the storage administrator can provide a compiled ACS REXX exit by:

1. Writing the REXX source program
2. Compiling the program
3. Copying the compiled REXX exit to the *fileid*:

```
IGDACSMC DFSMS VMSYS:DFSMS.ACSEXITS
```

activating the exit as described in [“Activation of the REXX Exit” on page 192](#)

For additional information and requirements for the REXX/370 compiler, refer to the *Licensed Program Specifications* (GH19-8161). Also, there are differences in the REXX implementations used by the Interpreter and the Compiler. For further information, refer to the REXX publications listed in [“Bibliography” on page 269](#).

Using Interpreted REXX

If you want to use the REXX ACS exit, but do not want to use compiled REXX, store your EXEC in IGDACSMC DFSMS VMSYS:DFSMS.ACSEXITS. When using an interpreted REXX ACS exit, you may experience performance problems that are not present when using compiled REXX.

Activation of the REXX Exit

DFSMS/VM allows activation of an ACS REXX exit when a source configuration file is activated. The REXX exit can be changed or removed while DFSMS/VM is running by changing or removing IGDACSMC DFSMS VMSYS:DFSMS.ACSEXITS and then reactivating.

Remember, placing your REXX exit in IGDACSMC DFSMS VMSYS:DFSMS.ACSEXITS is not enough. For the REXX exit to be used, you must also activate a configuration after you have placed your REXX exit there. Conversely, to remove an exit you must erase IGDACSMC DFSMS VMSYS:DFSMS.ACSEXITS and reactivate. DFSMS/VM remembers what exits were in use at the last activation. When you shutdown and restart, DFSMS/VM uses what was active at the time of shutdown.

The REXX exit is not contained in, nor specified by, a particular source configuration file, but may be activated whenever any configuration is activated. The REXX exit differs in this respect from the ACS routine which is assigned to a particular source configuration file during the translation process.

A source configuration file is activated by the DFSMS ACTIVATE command (see [“DFSMS ACTIVATE” on page 135](#)), the ISMF ACTIVATE command, or from ISMF by selecting option 5 on the Configuration Application Selection panel. If the storage administrator takes one of these actions, DFSMS/VM attempts to load IGDACSMC DFSMS VMSYS:DFSMS.ACSEXITS into storage and prepare it for execution. A message indicating the presence of a REXX exit is issued on the CONFIRM ACTIVATE REQUEST panel or after the DFSMS ACTIVATE command is issued. If you attempted to activate a configuration and did not receive this message a problem exists.

If the activated REXX exit causes severe problems (for example, the REXX exit rejects creation of required files or causes infinite loops), you can stop DFSMS/VM, erase DFSMS's copy of the REXX exit located in IGDACSMC DFSMS within the SFS directory VMSYS:DFSMS.ACTIVEACSEXITS, and then restart DFSMS/VM. You then need to activate a new configuration that has removed the earlier problems.

Coding the REXX Exit

This section provides information and suggestions to assist the storage administrator in writing the ACS REXX exit.

ACS Variables

The REXX exit uses the same read-only (R-O) and read/write (R/W) ACS variables as the ACS routine. However, the variable names used by the REXX exit do not contain a leading '&' as contained in the variable names of the ACS routine. For example, the ACS variable '&DIR' is represented as 'DIR' in a REXX exit. All variable names listed in this section are reserved and should not be used for other purposes by the REXX exit. Refer to [“ACS Language Read-Only Variables” on page 179](#) for additional information on R-O variables and [“ACS Language Read/Write Variables” on page 178](#) for R/W variables.

Read-Only Variables

The following read-only REXX variables are available for use in comparisons or other REXX exit operations:

- ACSENVIR
- DIR
- FN
- FPOOLID
- FT
- NDIRS
- OWNER
- PARENTMC
- PATH
- PATH.
- RECFM
- SIZE
- USER

These REXX variables are initialized by the EXTRACT subcommand (see “EXTRACT” on page 195). Although it is possible to modify the values of these variables in the REXX exit, any modifications are not saved after control is returned to DFSMS/VM.

With the exceptions noted below, the descriptions and initial values of these variables are identical to those provided for the ACS routine.

NDIRS and SIZE

These variables have numeric values in the ACS language. In the ACS REXX exit, they are represented by a character string of decimal digits. Arithmetic and comparison operations may be performed on these values using REXX operators.

If the file size is known, the input variable SIZE contains the file size in KB. If the file size is not known at the time ACS processing begins, SIZE is initialized to zero. Internally, DFSMS/VM is capable of storing a maximum file size of 21 4748 3647 KB. However, the default precision for REXX is nine digits. The storage administrator can use the REXX NUMERIC DIGITS instruction to increase the precision in the REXX exit to ten digits, thereby achieving the same precision used in the ACS routine.

PATH and PATH.

The ACS routine receives an input variable &PATH that is indexed to obtain the various directory levels. A similar capability is provided to the ACS REXX exit by the use of a simple variable (PATH) and a stem variable (PATH.). These variables are initialized by the EXTRACT subcommand as follows:

- PATH is initialized with the same value as the variable &PATH in the ACS routine. That is, it is the full name, excluding the file pool ID, of the directory which contains the file or directory being processed.
- The stem PATH. is initialized to the NULL string. The compound variables PATH.1 through PATH.NDIRS are then initialized to the corresponding directory levels.

For example, if automatic class selection is performed on a file contained in the directory VMSYSU:FRED.FINANCES.TAXES, the EXTRACT subcommand initializes variables as follows:

Example:

```
NDIRS  = 3
PATH   = 'FRED.FINANCES.TAXES'
PATH.1 = 'FRED'
PATH.2 = 'FINANCES'
PATH.3 = 'TAXES'
PATH.n = ''
```

/* where n is any integer > 3 */

Read/Write Variable

In DFSMS/VM, the only supported read/write variable is MGMTCLAS. The EXTRACT subcommand initializes the MGMTCLAS variable with the management-class name which is stored in DFSMS/VM after completion of the ACS routine. If the active configuration does not contain an ACS routine, but a REXX exit is active, MGMTCLAS is initialized to the value which would have been provided to the ACS routine if such a routine were active.

The MGMTCLAS variable can be used in comparison operations and is assigned a value by the REXX exit. If an assignment is performed, the normal REXX rules for assignment and variable substitution apply. The REXX exit saves the value of the MGMTCLAS variable to DFSMS/VM using the SET subcommand (see [“SET” on page 196](#)).

Other ACS Variables

Because MVS/DFP™ does not support REXX for automatic class selection, those ACS variables which are used in MVS/DFP but not in DFSMS/VM are not initialized by the EXTRACT subcommand. However, IBM recommends not to use these variable names.

The following lists the read-only variables supported by MVS/DFP that are not used or initialized by DFSMS/VM.

ACCT_JOB
ACCT_STEP
ALLVOL
ANYVOL
APPLIC
DD
DEF_DATACLAS
DEF_MGMTCLAS
DEF_STORCLAS
DSN
DSNTYPE
DSORG
DSOWNER
DSTYPE
EXPDT
GROUP
HLQ
JOB
LLQ
MAXSIZE
MEMHLQ
MEMNQUAL
MEMLLQ
MEMMN
MSVGP
NQUAL
NVOL
PGM
RECORG
RETPD
UNIT
XMODE

The following lists the read/write variables supported by MVS/DFP that are not used or initialized by DFSMS/VM.

STORCLAS
DATACLAS
STORGRP

DFSMS Subcommand Environment

DFSMS/VM establishes a subcommand environment called DFSMS for use by the ACS REXX exit. When the REXX exit is invoked, the default command destination is DFSMS, and commands issued from the REXX exit are routed to DFSMS.

The storage administrator can alter the command destination, either temporarily or permanently, from within the REXX exit using the ADDRESS instruction of REXX. One use for this is to allow the REXX exit to issue CMS or CP commands. If the command destination is permanently modified, it is necessary to use the ADDRESS instruction to direct any DFSMS subcommands to the DFSMS environment.

DFSMS Subcommands

DFSMS/VM provides subcommands that enable the ACS REXX exit to perform the same functions as the ACS routine. When issued from the ACS REXX exit, these subcommands are directed to DFSMS, unless the default command destination has been altered by the REXX exit. If this destination has been altered, each DFSMS subcommand must be preceded by ADDRESS DFSMS.

The DFSMS subcommands produce a return code of zero, if processing is normal. If an error is encountered during subcommand processing, a nonzero code is returned to the REXX exit. The REXX exit should check the return code from each subcommand using the special REXX variable RC and should take appropriate action if an error is detected.

Common return codes for all DFSMS subcommands are:

Return Code

Explanation

0

Processing was successful

4

An internal error occurred in DFSMS/VM

8

An unknown subcommand was passed to DFSMS/VM

Return codes for specific commands are listed after each command.

In the format diagrams that follow, the subcommand name is capitalized to indicate that it is a keyword and should not be used for other purposes. These keywords are not case-sensitive, that is, they can be issued using upper- or lower-case characters.

EXTRACT

The EXTRACT subcommand obtains values of ACS variables from DFSMS/VM storage and initializes the corresponding variables in the REXX exit.

The format of the EXTRACT subcommand is:

►► EXTRACT ◄◄

The R-O and R/W variables initialized by this subcommand are described in “ACS Variables” on [page 192](#). Normally the EXTRACT subcommand is issued only once in the ACS REXX exit. If this subcommand is issued more than once, each EXTRACT clause causes the ACS variables to be assigned the initial values.

When the EXTRACT subcommand is issued, the R/W variable MGMTCLAS is initialized with the value set by the ACS routine, if such a routine is active. If the active configuration does not contain an ACS routine,

or the ACS routine has not set the management class, the EXTRACT subcommand initializes MGMTCLAS as follows:

- For file or directory creation, the initial value assigned to MGMTCLAS is NULL.
- For file recall, the initial value assigned to MGMTCLAS is:
 - NULL, when a migrated file has the NULL management class or no management class
 - The file's current management class name when a migrated file has a management class name
- For conversion processing, the initial value assigned to MGMTCLAS is:
 - NULL, when the DFSMS MANAGE, MIGRATE, or CONVERT command processes a file or directory with no management class
 - NULL, when the DFSMS CONVERT command with the REDETERMINE option processes a file or directory that has the NULL management class
 - The file's current management class name when the DFSMS CONVERT command with the option REDETERMINE processes a file or directory which already has a management class name

In addition to the common return codes identified in [“DFSMS Subcommands” on page 195](#), the EXTRACT subcommand has the following return code:

Return Code	Explanation
-------------	-------------

12	You have specified an operand on the EXTRACT subcommand
----	---

SET

The SET subcommand saves the selected management class of the file or directory for which ACS processing is being performed.

The format of the SET subcommand is:

➤ SET ➤

When this subcommand is issued, DFSMS/VM commits the value of the REXX variable MGMTCLAS to DFSMS for that file or directory.

To modify the management class value for a file or directory, the variable MGMTCLAS must first be given a new value using a REXX assignment. This value must meet one of the following requirements or the SET subcommand fails:

- MGMTCLAS must meet the criteria described in [“Management Class Name” on page 39](#).

Note: If the MGMTCLAS variable is assigned a value that contains blanks, including leading or trailing blanks, the SET command fails. The REXX STRIP function can be used to trim blanks.

- MGMTCLAS can be assigned the NULL string ("). If this is done, SET assigns the NULL management class to the file or directory.

In addition, the storage administrator must be careful not to set a management class name that is not defined in the active configuration. If this situation occurs, the SET subcommand processor does not detect the error. However, the final step in ACS processing involves validation of the selected management class. If this management class is not defined in the active configuration, the ACS request fails.

Normally the SET subcommand should be issued only once during the ACS REXX exit routine, although multiple assignments can be made to the MGMTCLAS variable. If SET is issued more than once during the REXX exit, the management class value stored in DFSMS/VM for that file or directory is updated with each SET clause.

In addition to the common return codes identified in [“DFSMS Subcommands” on page 195](#), the SET subcommand has the following return codes:

Return Code	Explanation
-------------	-------------

12	You have specified an operand on the SET subcommand
----	---

16	The variable MGMTCLAS is not initialized or contains a value that is not an allowable management class name
----	---

WRITE

The WRITE subcommand is used for writing messages to the DFSMS/VM log and is functionally equivalent to the WRITE statement of the ACS language. These messages are logged only if the DFSMS/VM control file allows logging of informational messages.

The format of the WRITE subcommand is:

➤ WRITE — *expression* ➤

The *expression* variable is evaluated by REXX before it is passed to the subcommand processor. The string that results from evaluation of *expression* is written to the DFSMS/VM log if DFSMS/VM is configured to allow logging of informational messages. If the resulting string is longer than 250 characters, it is truncated at 250 characters.

The WRITE subcommand can log up to six messages from the ACS REXX exit. If the WRITE subcommand is issued after six messages are logged, an error code is returned.

Example:

```
write Roses are red
```

Assuming that the variables ROSES, ARE, and RED have not been assigned values, the following message is logged: Roses are red

Example:

```
color = 'blue'  
write 'Violets are' color
```

The following message is logged: Violets are blue

Note: If errors are detected during processing of the REXX exit, you can write error messages to the DFSMS/VM log.

In addition to the common return codes identified in [“DFSMS Subcommands” on page 195](#), the WRITE subcommand has the following return codes:

Return Code	Explanation
-------------	-------------

20	The length of the message exceeded the maximum and has been truncated (this is a warning)
----	---

24	The maximum number of messages has been exceeded
----	--

Returning Control to DFSMS/VM

At completion of the ACS REXX exit, control is returned to DFSMS/VM via the REXX RETURN (or EXIT) instruction. Refer to [z/VM: REXX/VM Reference](#) for a comparison of these instructions. These instructions

allow a result to be evaluated and passed back to the calling program. DFSMS/VM interprets these results as follows:

- If a result of zero is returned and the SET subcommand has been successfully issued from the REXX exit, DFSMS/VM passes the management class saved by the SET subcommand to the next ACS processing step.
- If a result of zero is returned and the SET subcommand has not been successfully issued, the management-class name of the file or directory is not changed. That is, the management-class name available as input to the REXX exit is passed to the next ACS processing step.
- If a positive integer is returned, DFSMS/VM rejects the ACS request for this file or directory. This means that the create, recall, migrate, or convert function that led to ACS invocation is not performed for this file or directory. Use this capability with caution as you can prevent general users from migrating files, creating files and directories, or from accessing their migrated files.
- If no result is returned, DFSMS/VM assigns a management class as if a result of zero were returned.
- Returning a negative integer from the REXX exit is an error and causes DFSMS/VM to fail ACS processing.
- Returning a noninteger result is an error and produces unpredictable results.

Recommendation: IBM recommends that a return code of zero be returned even if errors occur. With a return code of zero, general users can continue to create, migrate, and recall files. The WRITE command can ([“WRITE” on page 197](#)) be used to log error messages in to the DFSMS/VM log. The log needs to be reviewed periodically to see the type of problems impacting the general user community.

Handling of Errors

The storage administrator should use the REXX SIGNAL instruction for trapping events such as syntax errors and program interruptions. The return code should also be checked after subcommands are issued or external routines are called.

If there are errors when running the REXX exit and the errors are not trapped, control can be returned to DFSMS/VM with a positive integer. The result of this action is the rejection of the ACS request, see [“Returning Control to DFSMS/VM” on page 197](#).

If an error is encountered, the REXX exit should take appropriate action which includes:

- Notifying the storage administrator
- Logging the error (using the WRITE subcommand)
- Handling the ACS request by doing one of the following:
 - Retaining the input management class.
 - Assigning a default management class.
 - Assigning the NULL management class.
 - Rejecting the ACS request by returning a positive integer. This should be done with caution, because you can prevent general users from migrating files, creating files and directories, or from accessing their migrated files.

Before the ACS REXX exit is invoked, a recovery environment is established for handling abends. If the recovery environment is entered, DFSMS/VM takes the following action:

- An entry is created in the DFSMS/VM log
- ACS processing fails

Performance

Performance of the REXX exit is an important programming consideration, since this routine is invoked during any of the following situations:

- Creation of files (unless the Inherit or Defer options are selected for file creation) and directories

- Recall processing
- Conversion of files and directories (which can occur as a result of a DFSMS MANAGE, MIGRATE, or CONVERT command)

The storage administrator can improve the performance of the ACS REXX exit by:

- Using a compiled REXX exit
- Eliminating unnecessary calls to external programs
- Avoiding overuse of the DFSMS subcommands, for example, multiple uses of SET
- Exercising great care to eliminate the possibility of an infinite loop orabend
- Not performing I/O

Additional information and suggestions regarding performance of REXX programs is found in the following references:

- [z/VM: REXX/VM Reference](#)
- *REXX/370 Compiler: General Information*
- *REXX/370 Compiler: User's Guide and Reference*

Sample ACS REXX Routine for a Large System

The following ACS large system sample (see [Figure 102 on page 200](#)) is written in the REXX programming language.

```

/*****
/* EXEC NAME:  IGDACSMC DFSMS
/*****
/* Purpose:
/*   This EXEC provides examples for assigning management classes
/*   for large systems.  In compliance with the guidelines
/*   documented in the Installation and Customization Guide, the
/*   REXX SIGNAL instruction is used for trapping syntax errors.
/*   The return code is also checked after subcommands are issued
/*   and errors are recorded in the DFSMS/VM log.
/*
/*****
/* CHANGE HISTORY:
/*
/* mm/dd/yy Change by          Description
/* -----
/*
/*****
EXTRACT
if rc ^= 0 then signal Error_routine
rc = 0
signal on Syntax

if(acsenvir = 'RECALL') then do
/*****
/* Assign a management class for any file being RECALLED.
/*****
/* If the mgmtclas is null then
/*   - assign mgmtclas = 'TEMPFILE' for file type of 'TEMP' or file
/*   type beginning with 'LIST'.
/*   - assign mgmtclas = 'SQLFILE' for any file type with the
/*   string 'SQL' imbedded.
/*   - assign mgmtclas = 'PAYROLL' for any file with a parent
/*   management class that begins with 'PAYROLL'.
/*   - assign mgmtclas = 'STANDARD' for any management class = null
/*   that does not meet the prior criteria.
/*****
    if mgmtclas = '' then
        select
            when(ft = 'TEMP' | substr(ft,1,4) = 'LIST') then
                mgmtclas = 'TEMPFILE'
            when(pos('SQL',ft) ^= 0) then
                mgmtclas = 'SQLFILE'
            when(substr(parentmc,1,7) = 'PAYROLL') then
                mgmtclas = 'PAYROLL'
            otherwise
                mgmtclas = 'STANDARD'
        end
/* End Select */

```

Figure 102. Sample ACS REXX Routine for a Large System (Part 1 of 2)

```

/*****
/* If the size is greater than 500 kilobytes: */
/* - Do not change mgmtclas for files assigned a management class */
/* 'SQLFILE'. */
/* - Do not change mgmtclas for files assigned a management class */
/* 'TEMPFILE'. */
/* - Change mgmtclas to 'MIGRFAST' for files with a management */
/* class of 'PAYROLL'. */
/* - Assign mgmtclas = 'LDEFAULT' for any management class for any */
/* file with size > 500 kb that does not meet the prior */
/* criteria. */
*****/
if size > 500 then
  select
    when(mgmtclas = 'SQLFILE') then
      mgmtclas = 'SQLFILE'
    when(mgmtclas = 'TEMPFILE') then
      mgmtclas = 'TEMPFILE'
    when(mgmtclas = 'PAYROLL') then
      mgmtclas = 'MIGRFAST'
    otherwise
      mgmtclas = 'LDEFAULT'
  end
end
/* end select */
/* end do */

/*****
/* Assign a management class to directories when the ACS environment */
/* is CREATE or assign a management class to files and directories */
/* if the ACS environment is CONVERT. */
/* NOTE: MIGRATE and MANAGE will only do convert processing if */
/* a file or directory has the 'No' management class. */
*****/
if(acsenvir = 'CREATE' | acsenvir = 'CONVERT') then
  select
    when(ft = 'TEMP' | substr(ft,1,4) = 'LIST') then
      mgmtclas = 'TEMPFILE'
    when(pos('SQL',ft) <= 0) then
      mgmtclas = 'SQLFILE'
    when(dir = 'PAYROLL' & path = '') then
      mgmtclas = 'PAYROLL'
    otherwise
      if(substr(parentmc,1,7) = 'PAYROLL') then
        mgmtclas = 'PAYROLL'
      else
        mgmtclas = 'STANDARD'
      end
  end
/* end select */

SET
  if rc <= 0 then signal Error_routine
exit

/****Syntax Procedure*****/
/* Write syntax errors to the DFSMS/VM log. */
/*****/
Syntax:
  mgmtclas=''
  WRITE 'Invalid syntax from command:'
  WRITE ' ' sourceline(sig1)
  WRITE 'at line' sig1'.'
  SET
exit

/****Error_Routine Procedure*****/
/* Write to the DFSMS/VM log for commands receiving a non-zero rc */
/*****/
Error_routine:
  mgmtclas=''
  WRITE 'Unexpected return code' rc 'from command:'
  WRITE ' ' sourceline(sig1)
  WRITE 'at line' sig1'.'
  SET
exit

```

Figure 103. Sample ACS REXX Routine for a Large System (Part 2 of 2)

Appendix B. ACS Module Exit

PI

This appendix contains Programming Interface Information.

General Information

DFSMS/VM allows an installation-written module exit to take control after the ACS routine or ACS REXX exit or both have finished processing. Alternatively, the ACS module exit can be used instead of the ACS routine or ACS REXX exit to do the following:

- Modify management classes.
- Log customized informational messages. If you want informational messages displayed on the console or logged to the DFSMS log file, set the values for DFSMS_LOG_TO_CONSOLE and DFSMS_LOG_TO_FILE in the DFSMS control file to 4.
- Reject ACS processing. If the ACS module exit rejects ACS processing, the file is not migrated or recalled, or the directory is not created. If a DFSMS CONVERT or MANAGE command has been issued, no changes are made to the management class of the file or directory currently processing.

The ACS module exit can invoke the ACS routine to determine a new management class. The ACS module exit may also modify the ACS read-only and read/write variables and then invoke the ACS routine with the modified values. The management class obtained from reinvocation of the ACS routine may be assigned to the file or directory or used for comparison with the value provided as input to the module exit routine. The ACS module exit routine uses the ACS interface routine to reinvoke the ACS routine, refer to [“Invoking the ACS Routine”](#) on page 213 for additional information.

Note: Reinvoking the ACS routine does not cause the invocation of the REXX exit.

The module exit has the ability to call external programs and perform other tasks which can not be performed by the ACS routine. However, these tasks can generally be performed by the ACS REXX exit. The ACS routine and the REXX exit are easier to code and maintain than the module exit, and should be used when possible.

The DFSMS/VM ACS module exit is very similar to the management-class ACS installation exit of MVS/DFP. Unlike its MVS counterpart, however, the DFSMS/VM module exit can be dynamically activated (see [“Activation of the Module Exit”](#) on page 204).

Every effort must be made to avoid infinite loops, invocation of external routines that can abend, assignment of invalid management classes, or other situations that degrade the performance or reliability of the system. For example, invoking ACS on a file creation with an ACS module exit that creates a file in a file pool managed by DFSMS/VM is a scenario that causes an infinite loop.

Attributes of the Module Exit

The routine which is coded for the ACS module exit must have the following attributes:

- The module is a module file (IGDACSMC MODULE) which resides in the VMSYS:DFSMS.ACSEXITS directory.
- The routine is written in either High Level Assembler or Assembler H.
- The module is reentrant (capable of concurrent use by two or more tasks).
- The module must satisfy one of the following requirements:
 - The module is an ADCON-free module. This implies that the module runs correctly when loaded at an address different from that at which it was generated.
 - The module is a CMS module file that has relocation information saved. This can be done by using the CMS LOAD command with the RLDSAVE option, followed by the CMS GENMOD command.

- The routine has AMODE 31.
- The routine has RMODE ANY.

Activation of the Module Exit

DFSMS/VM allows an ACS module exit to be activated at the time a source configuration file is activated. The ACS module exit can be changed or removed while DFSMS/VM is running by changing or removing the IGDACSMC MODULE VMSYS:DFSMS.ACSEXITS and reactivating.

The module exit is not contained in, nor specified by, a particular source configuration file, but is activated whenever any configuration is activated. The exit differs in this respect from the ACS routine which is assigned to a particular source configuration file during the translation process.

To activate the ACS module exit it must be copied to *fileid*:

```
IGDACSMC MODULE VMSYS:DFSMS.ACSEXITS
```

before the ACTIVATE command is invoked.

A source configuration file is activated by the DFSMS ACTIVATE command (see “DFSMS ACTIVATE” on page 135), the ISMF ACTIVATE command, or from ISMF by selecting option 5 on the Configuration Application Selection panel. If the storage administrator takes one of these actions, DFSMS/VM attempts to load IGDACSMC MODULE VMSYS:DFSMS.ACSEXITS into storage and prepare it for execution. A message indicating the presence of a module exit is issued on the Confirm Activate Request panel or after the DFSMS ACTIVATE command is issued. If you attempted to activate a configuration and did not receive this message a problem exists.

Warning: The storage administrator, unless necessary, should not access a minidisk or directory in the virtual machines which perform ACS processing (DFSMS master and server machines). If this is done and the accessed minidisk or directory contains a file IGDACSMC MODULE, this module could be installed as the exit rather than the module contained in directory VMSYS:DFSMS.ACSEXITS.

Remember, placing your ACS module exit in IGDACSMC MODULE VMSYS:DFSMS.ACSEXITS is not enough. For the module exit to be used, you must also activate a configuration after you have placed your module exit there. Conversely, to remove an exit, you must erase IGDACSMC MODULE VMSYS:DFSMS.ACSEXITS and reactivate. DFSMS remembers what exits were in use at the last activation. When you shutdown and restart, DFSMS uses what was active at the time of shutdown.

If a problem occurs in the module exit, you need to remove and modify the ACS module exit located in IGDACSMC MODULE VMSYS:DFSMS.ACSEXITS and then reactivate the configuration.

Coding the Module Exit

This section provides information to assist the storage administrator in writing the ACS module exit.

Entry Parameters

Figure 104 on page 205 illustrates the parameter structure upon entry to the ACS module exit. A 4 KB work area on a doubleword boundary is reenterable. The following macros (found in FSMPSI MACLIB are stored on the BUILD disk) map the parameters that are passed to the ACS module exit:

IGDACSPM

Describes the parameter list for the ACS module exit. See “Parameter List for the ACS Module Exit (IGDACSPM)” on page 207.

IGDACERO

Maps the data area pointed to by ACSPERO. This area contains the input values of the read-only and read/write variables that the ACS module exit routine can reference when selecting a management class. For compatibility with MVS, this data area includes fields for MVS variables that are not used in DFSMS/VM. See “Input Variables for the ACS Module Exit (IGDACERO)” on page 208.

IGDACERW

Maps the data area in which the exit can save a management-class name and message information to be returned to DFSMS/VM. See [“Values Set by the ACS Module Exit \(IGDACERW\)”](#) on page 213.

Note: To use any of the entry parameters constants when coding the module exit, you must identify the macro statement that includes the constant with a label of the CSECT name of the exit.

DFSMS/VM provides an interface routine which the module exit can use to reinvoke the ACS routine. The parameter list that is passed to the module exit, as shown in [Figure 104](#) on page 205, contains a field (ACSPACS) which points to the ACS interface routine. For more information, see [“Invoking the ACS Routine”](#) on page 213.

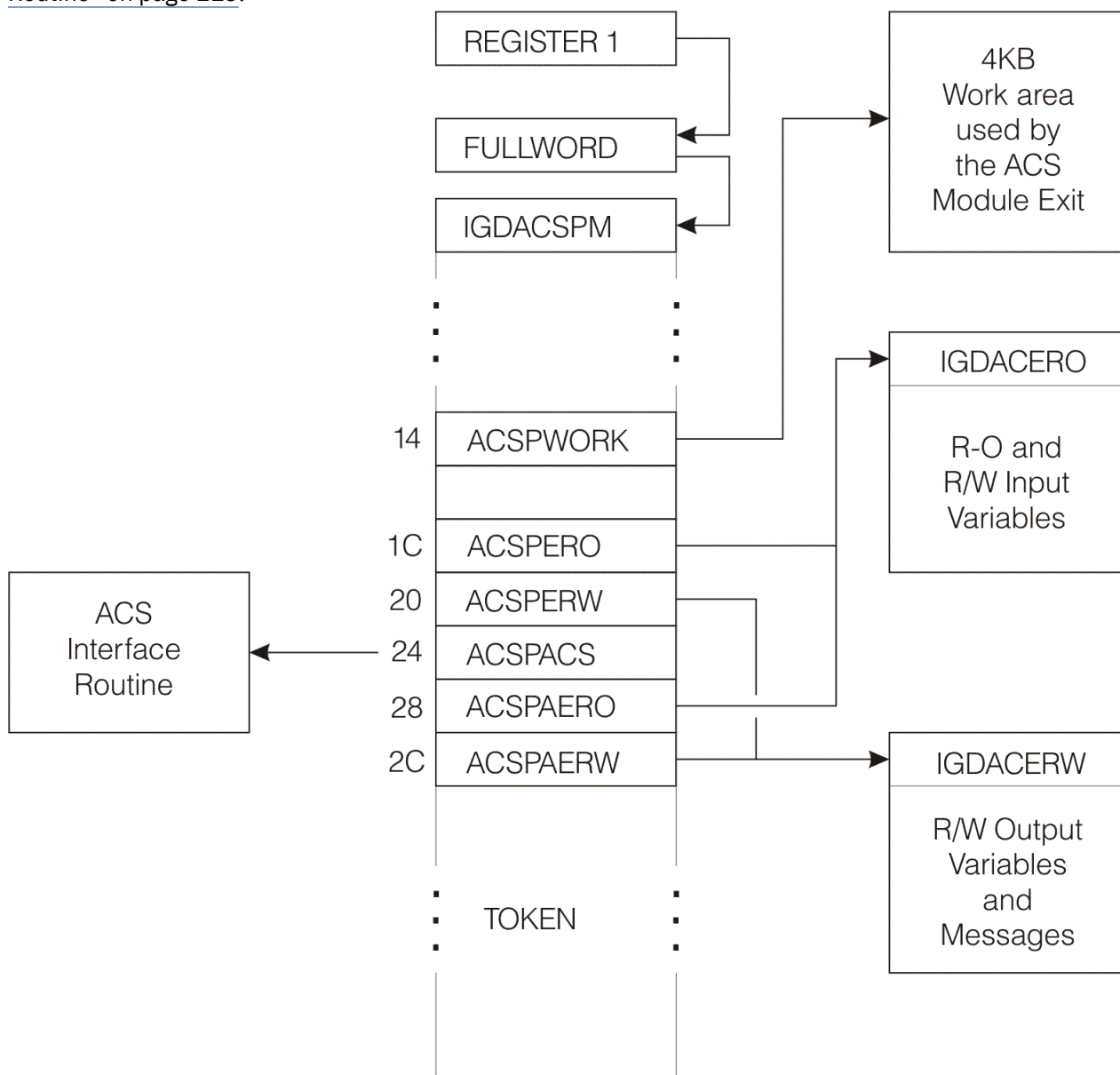


Figure 104. Parameter Structure for the ACS Module Exit

Assigning a Management Class

Upon entry to the ACS module exit, ACSPERW points to a data area that has been initialized to binary zeros. To assign a management class, the exit must map this data area with macro IGDACERW and set the following fields:

- Set the ACERWNCS field to 1. This field specifies the number of management classes to be assigned by the module exit. The valid values for this field are 0 and 1. Initially, it contains 0, indicating that the module exit has not assigned a management class.
- Set the ACERWVLN field to the length of the management-class name which is to be assigned. This length must be in the range 0–8, inclusive.
- Set the ACERWVAL field to the name of the management class to be assigned.

To assign a NULL management class, the exit must set the ACERWNCS field to 1 and the ACERWVLN field to 0. The ACERWVAL field is then ignored by DFSMS/VM. If the ACS module exit does not assign a management class, field ACERWNCS should be left as 0. If this is done, DFSMS/VM assigns the management class that was provided upon entry to the module exit.

Returning Messages

An ACS module exit can return messages that are written to the DFSMS/VM log, provided that the DFSMS control file allows logging of informational messages. To return messages, the exit must set ACERWNMG equal to the number of messages to be returned and place the text of the messages in ACERWMSG. ACERWMSG can hold up to six messages. Messages must be 250 bytes long and should be padded with blanks, if necessary.

Returning Control to DFSMS/VM

Register contents

Upon return from the ACS module exit, register contents must be as follows:

- Register 0 may contain a reason code
- Registers 1–14 must be restored to their entry values
- Register 15 contains a return code

Return codes

The ACS module exit must terminate with a return code in register 15 that indicates what action is to be taken upon return from the exit. The return codes and their meanings are described below:

Code

Meaning

00(X'00')

Indicates processing completed normally, and that the management class stored by the module exit is to be used.

04(X'04')

Indicates that the create, recall, or convert request that led to ACS invocation should be rejected and that register 0 contains an installation-supplied reason code.

Note: If 04 is returned, DFSMS/VM rejects the ACS request for this file or directory. This means that the create, recall, or convert that led to ACS invocation will not be performed for this file or directory. This capability must be used carefully, you can prevent general users from creating files and directories, migrating files, or from being able to access their migrated files.

16(X'10')

Indicates that the ACS module exit contains an error and should be disabled (deactivated). The module exit can be reactivated by activating a source configuration file. See [“Activation of the Module Exit” on page 204](#).

Any other return code represents an error.

Reason Codes

The ACS module exit can load a reason code in register 0. The installation determines the reason codes and their meanings. When the exit passes back a return code of X'04' in register 15, the reason code contained in register 0 appears in the text of a message issued by DFSMS/VM.

ACS Module Exit Errors

The module exit is deactivated if the module exit returns an undefined return code or a return code of 16, or if an abend occurs in the module exit and it is handled by DFSMS/VM. DFSMS/VM provides error recovery to regain control if a module exit abends. If the exit needs to free resources in case of errors, it must provide its own error recovery, and ensure that this error recovery environment is deleted before returning to DFSMS/VM.

Management Class Variables

Management class type variables (management class or parent management class) in the provided mapping macros have a length field and a 30-character field. A length of zero indicates the NULL management class (for input management classes, this could also indicate no management class). Management classes used under CMS have a maximum length of eight. (You must be careful when assigning management classes, see, “[Assigning a Management Class](#)” on page 205.). The length variable ACEROMCL indicates the number of valid characters. For example, if ACEROMCL equals four and ACEROMCV equals *BLUEnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn*, the management class is *BUE* (the first four characters); the remainder of the string filled in by the character *n* provides the length of the variable.

Parameter List for the ACS Module Exit (IGDACSPM)

When the ACS module exit is invoked, Register 1 contains a pointer to the address of the structure shown in Figure 105 on page 208. This linkage is shown graphically in Figure 104 on page 205.

Offsets	Type	Length	Name	Description	
Decimal	Hex				
0	(0)	CHARACTER	52	ACSPM(0)	ACS EXIT PARAMETER LIST
0	(0)	CHARACTER	8	ACSPID	CONTROL BLOCK ID='IGDACSPM'
8	(8)	SIGNED	2	ACSPLN	ACSP CONTROL BLOCK LENGTH
10	(A)	SIGNED	2	ACSPVER	CONTROL BLOCK VERSION
12	(C)	CHARACTER	8		RESERVED
20	(14)	ADDRESS	4	ACSPWORK	POINTER TO A WORK AREA FOR THE EXIT
24	(18)	SIGNED	4	ACSPWLEN	LENGTH OF WORK AREA
28	(1C)	ADDRESS	4	ACSPERO	POINTER TO READ ONLY VARIABLES MAPPED BY IGDACERO.
32	(20)	ADDRESS	4	ACSPERW	POINTER TO READ/WRITE VARIABLES MAPPED BY IGDACERW
36	(24)	ADDRESS	4	ACSPACS	POINTER TO INTERFACE ROUTINE FOR INVOKING THE ACS ROUTINES LINKAGE IS VIA STANDARD LINKAGE CONVENTIONS. IF THIS FIELD IS ZERO, THEN NO ACS ROUTINE EXISTS FOR THE CURRENT CONSTRUCT.
40	(28)	CHARACTER	12	ACSPACSP	PARAMETERS FOR ACS ROUTINES
40	(28)	ADDRESS	4	ACSPAERO	POINTER TO READ ONLY VARIABLES INITIALLY SET TO ACSPERO.
44	(2C)	ADDRESS	4	ACSPAERW	POINTER TO READ/WRITE VARS, INITIALLY SET TO ACSPERW.
48	(30)	CHARACTER	4	ACSPATOK	TOKEN FOR USE BY ACS INTERFACE ROUTINE (DO NOT MODIFY)
=====					
CONSTANT FOR THE VERSION NUMBER					
=====					
ACSVER		0	VERSION NUMBER		
=====					
RETURN CODES FROM SMS INSTALLATION EXIT					
=====					
ACSPCOMP		0	EXIT COMPLETED SUCCESSFULLY		
ACSJERR		4	ERROR, REJECT THIS ACS REQUEST		
ACSEXERR		16	AN ERROR OCCURRED IN THE EXIT. DON'T REINVOKE IT		

Figure 105. Layout of ACS Module Exit Parameter List

Input Variables for the ACS Module Exit (IGDACERO)

The ACS read-only variables are mapped with the **IGDACERO** mapping macro. The layout of this macro is shown in [Figure 106 on page 209](#). All MVS-only variables are initialized to binary zero in DFSMS/VM.

Offsets	Type	Length	Name	MVS Only	Description
0	(0) CHARACTER	1780	ACERO(0)		ACS READ-ONLY VARIABLES
=====					
CONTROL BLOCK HEADER					
=====					
0	(0) CHARACTER	8	ACEROID		CONTROL BLOCK ID=ACERO
8	(8) SIGNED	2	ACEROLEN		LENGTH OF CONTROL BLOCK
10	(A) SIGNED	2	ACEROVER		CONTROL BLOCK VERSION
=====					
READ ONLY VARIABLES FOLLOW					
=====					
12	(C) SIGNED	4	ACEROSIZ		PRIMARY/ACTUAL SIZE OF DATA SET IN KB
16	(10) SIGNED	4	ACEROMSZ	X	MAXIMUM SIZE OF DATA SET IN KB
20	(14) CHARACTER	8	ACEROUNT	X	UNIT NAME
28	(1C) CHARACTER	8	ACEROMVG	X	MSS VOLUME GROUP NAME
36	(24) CHARACTER	8	ACEROAPP	X	APPLICATION ID (RACF)
44	(2C) CHARACTER	8	ACERODSO	X	DATA SET OWNER (RACF)
52	(34) CHARACTER	8	ACEROUSR		USERID
60	(3C) CHARACTER	8	ACEROGRP	X	GROUPID
68	(44) SIGNED	4	ACERODSG	X	DATA SET ORGANIZATION (MAY BE ACEROPS, ACEROPO, ACEROVS, ACERODA, ACERONUL)
72	(48) SIGNED	4	ACERORCG	X	DATA SET RECORD ORGANIZATION (MAY BE ACEROKS, ACEROES, ACERORR, ACEROLS, ACERONUL)
76	(4C) SIGNED	4	ACERODST	X	DATA SET TYPE (MAY BE ACEROGDS, ACEROPRM, ACEROTMP, ACERONUL)
80	(50) SIGNED	4	ACEROXMD	X	EXECUTION MODE (MAY BE ACEROBCH, ACEROTSO, ACEROTSK, ACERONUL)
84	(54) CHARACTER	8	ACEROJOB	X	JOB NAME
92	(5C) CHARACTER	8	ACERODD	X	DD NAME
100	(64) CHARACTER	8	ACEROPGM	X	PROGRAM NAME
108	(6C) CHARACTER	4	ACEROEXP	X	EXPIRATION DATE (YYYYDD) IN PACKED DECIMAL
112	(70) SIGNED	4	ACERORTP	X	RETENTION PERIOD DAYS
116	(74) CHARACTER	128			RESERVED

Figure 106. Layout of IGDACERO Mapping Macro (Part 1 of 4)

```

=====
ENVIRONMENT WHERE INVOKED - MUST BE:

    ACERORCL FOR RECALL,
    ACERORCV FOR RECOVER (MVS),
    ACEROCNV FOR CONVERT,
    ACEROALC FOR NEW ALLOCATIONS (MVS),
    ACEROSTE FOR OSMI STORE (MVS),
    ACEROCHE FOR OSMI CHANGE (MVS),
    ACEROCTE FOR OSMC CLASS TRANSITION (MVS),
    ACEROCRE FOR CREATION OF FILES OR DIRECTORIES (VM),

OR INSTALLATION EXIT MAY SET OWN VALUE.
=====

244  (F4) CHARACTER      8  ACEROENV
=====

DEFAULT VALUES OF READ/WRITE VARIABLES
=====

252  (FC) CHARACTER      32  ACERODDC  X  DEFAULT DATA CLASS (FROM RACF)
252  (FC) SIGNED         2  ACERODDL  X  LENGTH OF DEFAULT DATACLAS
254  (FE) CHARACTER      30  ACERODDV  X  VALUE OF DEFAULT DATACLAS
284  (11C) CHARACTER     32  ACERODSC  X  DEFAULT STORAGE CLASS (FROM
    RACF)
284  (11C) SIGNED        2  ACERODSL  X  LENGTH OF DEFAULT STORCLAS
286  (11E) CHARACTER     30  ACERODSV  X  VALUE OF DEFAULT STORCLAS
316  (13C) CHARACTER     32  ACERODMC  X  DEFAULT MANAGEMENT CLASS (FROM
    RACF)
316  (13C) SIGNED        2  ACERODML  X  LENGTH OF DEFAULT MGMTCLAS
318  (13E) CHARACTER     30  ACERODMV  X  VALUE OF DEFAULT MGMTCLAS
348  (15C) CHARACTER     80  RESERVED

=====

INPUT VALUES OF READ/WRITE VARIABLES
=====

428  (1AC) CHARACTER     32  ACERODC   X  DATA CLASS INPUT ONLY. OUTPUT
    RETURNED IN IGDACERW
428  (1AC) SIGNED        2  ACERODCL  X  LENGTH OF DATACLAS
430  (1AE) CHARACTER     30  ACERODCV  X  VALUE OF DATACLAS
460  (1CC) CHARACTER     32  ACEROSC   X  STORAGE CLASS INPUT ONLY.
    OUTPUT RETURNED IN IGDACERW
460  (1CC) SIGNED        2  ACEROSCL  X  LENGTH OF STORCLAS
462  (1CE) CHARACTER     30  ACEROSCV  X  VALUE OF STORCLAS
492  (1EC) CHARACTER     32  ACEROMC   X  MANAGMENT CLASS INPUT ONLY.
    OUTPUT RETURNED IN IGDACERW
492  (1EC) SIGNED        2  ACEROMCL  X  LENGTH OF MGMTCLAS
494  (1EE) CHARACTER     30  ACEROMCV  X  VALUE OF MGMTCLAS

=====

DATA SET NAME
=====

524  (20C) CHARACTER     44  ACERODSN  X  DATA SET NAME
568  (238) CHARACTER      8  RESERVED
576  (240) SIGNED        4  ACERODNT  X  DSNTYPE (MAY BE ACEROLIB
    ACEROPDS, ACERONUL)

```

Figure 107. Layout of IGDACERO Mapping Macro (Part 2 of 4)

```

=====
MVS ACCOUNTING INFORMATION
=====

580 (244) CHARACTER      257 ACEROJAC  X  JOB
580 (244) UNSIGNED       1 ACEROJNM  X  NUMBER OF FIELDS
581 (245) CHARACTER      256 ACEROJFL  X  FIELDS
837 (345) CHARACTER       7          RESERVED
844 (34C) CHARACTER      257 ACEROSAC  X  STEP
844 (34C) UNSIGNED       1 ACEROSNM  X  NUMBER OF FIELDS
845 (34D) CHARACTER      256 ACEROSFL  X  FIELDS
1101 (44D) CHARACTER       7          RESERVED

=====
VOLSER INFORMATION
=====

1108 (454) SIGNED         2 ACERONVL  X  NUMBER OF VOLSERS
1110 (456) CHARACTER      6 ACEROVOL(59) X DIMENSION=(59) ARRAY OF 6 BYTE
                                VOLSERS
1464 (5B8) CHARACTER      8          RESERVED FOR FUTURE EXTENSIONS

=====
MEMBER NAME
=====

1472 (5C0) CHARACTER      44 ACEROMEM  X  MEMBER NAME
1516 (5EC) CHARACTER      8          RESERVED

=====
VM SPECIFIC VARIABLES
=====

1524 (5F4) CHARACTER       8 ACEROFN    VM FILE NAME
1532 (5FC) CHARACTER       8 ACEROFT    VM FILE TYPE
1540 (604) CHARACTER       8 ACEROFPI   VM FILE POOL ID
1548 (60C) CHARACTER      144 ACEROPTH   PATH NAME
1692 (69C) CHARACTER      20          EXPANSION
1712 (6B0) CHARACTER      16 ACERODIR   DIRECTORY VARIABLE
1728 (6C0) CHARACTER       8 ACEROONR   ID OF OWNER
1736 (6C8) CHARACTER      32 ACEROPMC   PARENT MGMT CLASS
1736 (6C8) SIGNED         2 ACEROPML   LENGTH OF PARENT MGMTCLAS
1738 (6CA) CHARACTER      30 ACEROPMV   VALUE OF PARENT MGMTCLAS
1768 (6E8) CHARACTER       8          EXPANSION
1776 (6F0) CHARACTER       1 ACEROFMT   RECFM
1777 (6F1) CHARACTER       3          WORD BNDRY ALIGN

=====
CONSTANTS FOR THE FOLLOWING FIELDS IN ACERO:
ACERODSG, ACERORCG, ACERODST, ACEROXMD, AND
ACEROENV, AND HEADER AND NULL DEFINITIONS
=====

ACEROI      'ACERO '  CONTROL BLOCK ID
ACEROV      1         VERSION NUMBER
ACERONUL    '00'X    NULL

```

Figure 108. Layout of IGDACERO Mapping Macro (Part 3 of 4)

```

=====
Record Formats for VM
=====

ACEROFR      'F'      FIXED
ACEROVR      'V'      VARIABLE

=====
ACERODSG - DATA SET ORGANIZATION (MVS)
=====

ACEROPS      1          PS PHYSICAL SEQUENTIAL
ACEROPO      2          PO PARTITIONED
ACEROVS      3          VS VSAM ORGANIZATION
ACERODA      4          DA BDAM ORGANIZATION

=====
ACERORCG - DATA SET RECORD ORGANIZATION (MVS)
=====

ACEROKS      1          KSDS VSAM CLUSTER
ACEROES      2          ESDS VSAM ENTRY SEQUENCED
ACERORR      3          RRDS VSAM RELATIVE RECORD
ACEROLS      4          LINEAR SPACE

=====
ACERODST - DATA SET TYPE (MVS)
=====

ACEROGDS      1          ONE GENERATION DATA SET OF A GDG
ACEROPRM      2          STANDARD PERMANENT DATA SETS
ACEROTMP      3          TEMPORARY DATA SETS

=====
ACEROXMD - EXECUTION MODE (MVS)
=====

ACEROBCH      1          BATCH EXECUTION MODE
ACEROTSO      2          TSO EXECUTION MODE
ACEROTSK      3          STARTED TASK

=====
ACEROENV - ENVIRONMENT WHERE INVOKED
=====

ACEROSTE      'STORE'   ' OSMI STORE (MVS)
ACEROCH      'CHANGE'   ' OSMI CHANGE (MVS)
ACEROCTE      'CTANS'   ' OSMC CLASS TRANSITION (MVS)
ACERORCL      'RECALL'   ' DATA SET RECALL OPERATIONS
ACERORCV      'RECOVER'  ' DATA SET RECOVER OPERATIONS (MVS)
ACEROCNV      'CONVERT'  ' DATA SET CONVERT IN PLACE OPERATIONS
ACEROALC      'ALLOC'    ' NEW DATA SET ALLOCATIONS (MVS)
ACEROCRE      'CREATE'   ' VM FILE OR DIRECTORY

=====
ACERODNT - DSNTYPE (MVS)
=====

ACEROLIB      1          DSNTYPE=LIBRARY
ACEROPDS      2          DSNTYPE=PDS

```

Figure 109. Layout of IGDACERO Mapping Macro (Part 4 of 4)

Values Set by the ACS Module Exit (IGDACERW)

The ACS read/write variables are mapped with the **IGDACERW** mapping macro. The layout of this macro is shown in [Figure 110 on page 213](#).

Offsets	Type	Length	Name	Description
0	(0) CHARACTER	1172	ACERW(0)	ACS READ/WRITE VARIABLES
=====				
CONTROL BLOCK HEADER				
=====				
0	(0) CHARACTER	8	ACERWID	CONTROL BLOCK ID=ACERW
8	(8) SIGNED	2	ACERWLEN	LENGTH OF CONTROL BLOCK
10	(A) SIGNED	2	ACERWVER	CONTROL BLOCK VERSION
=====				
EXECUTOR FILLS IN THE FOLLOWING R/W VARIABLE FIELD(S)				
=====				
12	(C) SIGNED	4	ACERWNCS	NUMBER OF CLASS SELECTION
16	(10) CHARACTER	480	ACERWCSV(15)	RETURN VARIABLES DIMENSION=(15) CLASS SELECTION VARIABLE RETURNED
16	(10) SIGNED	2	ACERWVLN	LENGTH OF VALUE
18	(12) CHARACTER	30	ACERWVAL	VALUE RETURN VARIABLES
48	(30) CHARACTER	448		
496	(1F0) CHARACTER	2		RESERVED
=====				
MESSAGE AREA				
=====				
498	(1F2) SIGNED	2	ACERWNMG	NUMBER OF MESSAGES
500	(1F4) CHARACTER	1500	ACERWMSG(6)	DIMENSION=(6) MESSAGES GENERATED BY EXECUTION OF ACS ROUTINE WRITE STATEMENTS.
500	(1F4) CHARACTER	250	ACERWTXT	TEXT OF MESSAGES
750	(262) CHARACTER	1250		
2000	(488) CHARACTER	12		RESERVED
=====				
CONSTANTS FOR THE HEADER FIELDS				
=====				
ACERWI		'ACERW	'	CONTROL BLOCK ID
ACERWV		0		VERSION NUMBER

Figure 110. Layout of IGDACERW Mapping Macro

Invoking the ACS Routine

The ACS interface routine is used to invoke the ACS routine from the ACS module exit. The ACSPACS field shown in [Figure 104 on page 205](#) contains the address of this interface routine. If the ACSPACS field contains a zero, no ACS routine exists.

[Figure 111 on page 214](#) illustrates the parameter structure for the ACS interface routine. This parameter list, ACSPACSP, is imbedded in the parameter list that is passed to the ACS module exit. ACSPACSP contains the following fields:

ACSPAERO

Points to a list of ACS variables mapped by IGDACERO that are used by the ACS interface routine. Initially, ACSPAERO points to the same list of read-only and read/write variables as ACSPERO. The exit can modify the passed variables before invoking the ACS routine. Alternatively, the module exit routine can create an entirely new list of ACS variables and point to them with ACSPAERO before invoking the ACS interface routine. By modifying these variables and then invoking the ACS routine, the module exit routine can compare the original management-class name with the new name selected by the ACS routine.

ACSPAERW

ACSPAERW contains the same value as ACSPERW, a pointer to a data area initialized with binary zeros. When the ACS routine is executed, it sets the management-class and message fields pointed to by ACSPAERW. When the ACS module exit returns control to DFSMS/VM with a return code of zero,

the management class contained in the data area pointed to by ACSPERW is assigned to the file or directory for which automatic class selection was performed.

Note: If the ACS module exit places a new address in ACSPAERW, and the storage administrator intends for the values pointed to by ACSPAERW to be used by DFSMS/VM, it must copy these values into the area pointed to by ACSPERW. If this is not done, the correct values will not be used upon return to DFSMS/VM.

If the ACS routine selects a management class of NULL: ACERWNCS, ACERWVLN, and ACERWVAL are all set to zero. In order for the module exit to cause the selection of a management class of NULL, you must set the ACERWNCS field to 1 and leave the ACERWVLN and ACERWVAL fields at zero.

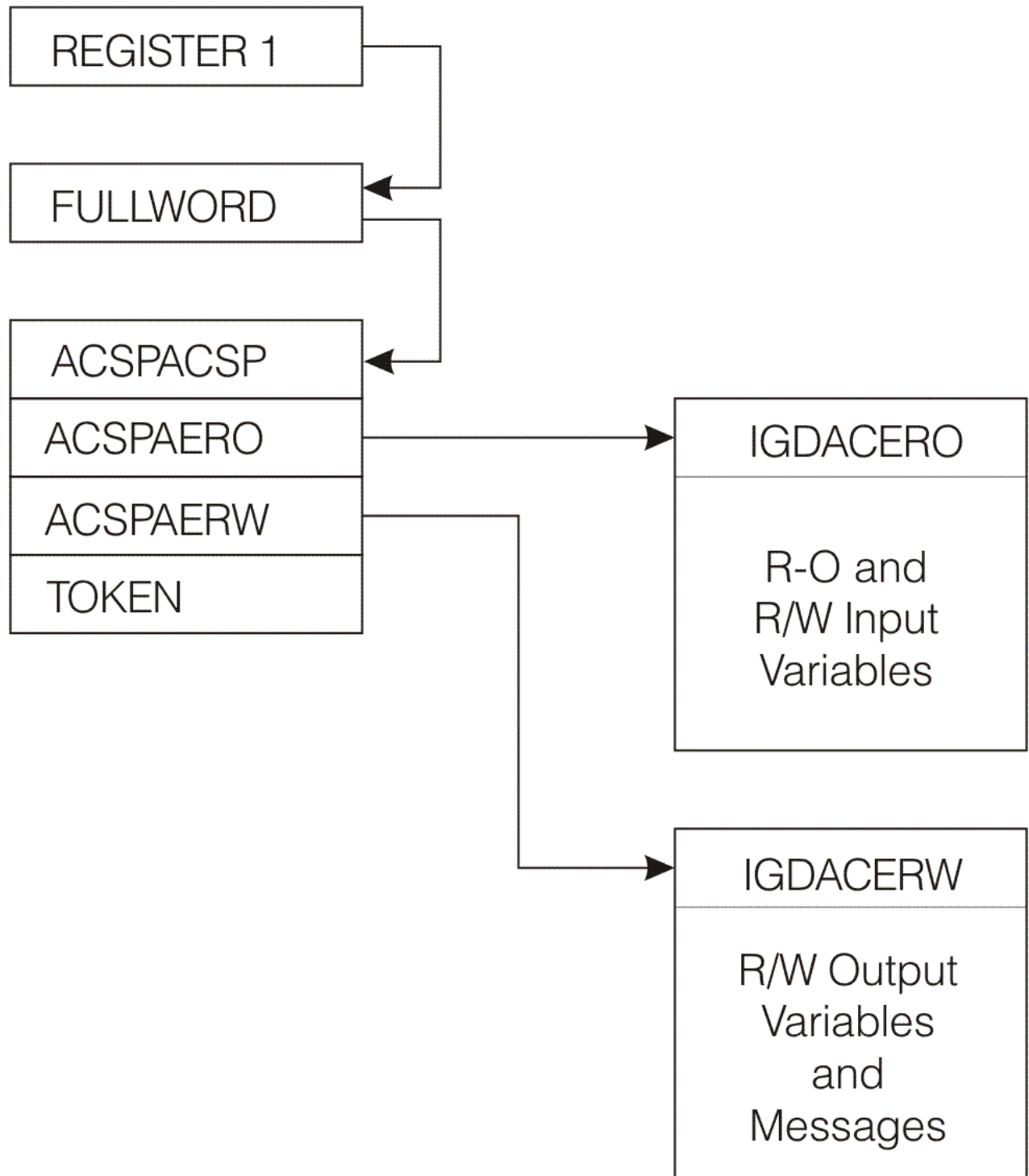


Figure 111. Parameter Structure for the ACS Interface Routine

Return and Reason Codes from the ACS Routine

During ACS exit processing, the module exit can reinvoke the main ACS processing by invoking the ACS interface routine. Return and reason codes may be returned from the interface routine to the module exit after ACS has been reinvoked. If the return code from the interface routine indicates failure, it does not necessarily mean that ACS processing will fail for the file. The return code from the module exit determines whether the processing will fail.

ACS Routine Return Codes

The ACS routine return codes (see [Table 7 on page 215](#)) may be found in register 15 upon return from the ACS interface routine after ACS is reinvoked.

Table 7. ACS Routine Return Codes	
Return Code	Meaning
0 (X'00')	Successful.
8 (X'08')	Invalid parameter list; register 0 contains the reason code.
12 (X'0C')	The main ACS routine set a nonzero exit code via the EXIT CODE statement; register 0 contains the exit code.
16 (X'10')	ACS internal error.
20 (X'14')	ACS internal error.
24 (X'18')	ACS internal error.

For information about ACS reason codes (register 0), refer to [z/VM: DFSMS/VM Messages and Codes](#) message IGD1022E.

Appendix C. Samples of DFSMS/VM Systems Applications

To demonstrate how you can apply the DFSMS/VM functions and commands to your system, this appendix provides two sample systems. One is a large system application of DFSMS/VM and the other is a small system application using DFSMS/VM. The large and small systems use sample ACS routines and source configuration files provided with the DFSMS/VM product tape. The following sections provide:

- Samples of management classes
- Use of an ACS routine
- Use of a configuration
- Results of converting a storage group
- Results of managing a storage group

The large system sample performs migration to ML1. In this sample, ACS for file creation is set to the INHERIT value. The source configuration file, SMPCNFGL CONFIG, and the ACS routine used by this configuration, SMPACSL ACS, are provided on the DFSMS/VM product tape. During installation, these files are stored on the SES LOCALSMP disk. SMPCNFGL CONFIG contains the management classes, ACS routine, and base configuration information for this sample system.

The small system sample does expiration only. In this sample, ACS for file creation is set to YES (invoking ACS for file creation is not recommended, but is included as an example). The source configuration file, SMPCNFGS CONFIG, and the ACS routine used by this configuration, SMPACSS ACS, are provided on the DFSMS/VM product tape. During installation, these files are stored on the SES LOCALSMP disk. Other samples like DGTVCNTL SAMPDATA and DGTVAUTH SAMPDATA as well as the profile exec for the SMSMASTR, FSMPROF EXEC, are also located on the SES LOCALSMP disk. SMPCNFGS CONFIG contains the management classes, ACS routine, and base configuration information for this sample system.

A sample ACS routine and configuration that does migration to ML1 and ML2, with ACS for file creation set to the DEFER value are also provided on the product tape. The sample ACS routine is SMPACS2 ACS and the sample source configuration file is SMPCNFG2 CONFIG.

There is also a sample REXX exit and configuration for large systems included on the DFSMS/VM product tape. The sample REXX exit is SMPREXX EXEC and the sample configuration file is SMPCNFGR CONFIG.

For our examples, the following management classes are used in the sample ACS routines.

Name

Characteristics

NOEXPIRE

No expiration (small system and systems migrating to ML1 and ML2).

Expire After Days Nonusage: NOLIMIT

Expire Since Days Creation: NOLIMIT

Primary Storage Days Nonusage: N/A

Secondary Storage Days Nonusage: N/A

Space Management Technique: NONE

SQLFILE

SQL files and directories no expiration and no migration (large system).

Expire After Days Nonusage: NOLIMIT

Expire Since Days Creation: NOLIMIT

Primary Storage Days Nonusage: N/A

Secondary Storage Days Nonusage: N/A

Space Management Technique: NONE

NEWUSER

Management class for a new user. Expiration one year from date of last reference and two years since creation date.

Expire After Days Nonusage: 366

Expire Since Days Creation: 731

Primary Storage Days Nonusage: N/A

Secondary Storage Days Nonusage: N/A

Space Management Technique: NONE

TEMPFILE

Expiration seven days from creation and no migration. This may be applied to files and directories that are only needed for a limited time or can easily be re-created (Class Exercise files, perhaps) (small system, large system, and systems migrating to ML1 and ML2)

Expire After Days Nonusage: NOLIMIT

Expire Since Days Creation: 7

Primary Storage Days Nonusage: N/A

Secondary Storage Days Nonusage: N/A

Space Management Technique: NONE

PAYROLL

No expiration and migration to ML1 21 days from the date of last reference. This may apply to files that can never be deleted for legal or other reasons (large system and systems migrating to ML1 and ML2)

Expire After Days Nonusage: NOLIMIT

Expire Since Days Creation: NOLIMIT

Primary Storage Days Nonusage: 21

Secondary Storage Days Nonusage: NOLIMIT

Space Management Technique: MIGRATE

STANDARD

Expiration two years from date of creation and six months from date of last reference, and migration to ML1 two weeks from the date of last reference. This can be applied to most normal files that have no special retention requirements (large system).

Expire After Days Nonusage: 183

Expire Since Days Creation: 731

Primary Storage Days Nonusage: 14

Secondary Storage Days Nonusage: NOLIMIT

Space Management Technique: MIGRATE

MIGRFAST

No expiration and migrate to ML1 four days from the date of last reference. This can be applied to very large files which are never deleted for legal or other reasons (large system).

Expire After Days Nonusage: NOLIMIT

Expire Since Days Creation: NOLIMIT

Primary Storage Days Nonusage: 4

Secondary Storage Days Nonusage: NOLIMIT

Space Management Technique: MIGRATE

LDEFAULT

Expiration two years from date of creation and six months from date of last reference, and migration to ML1 four days from the date of last reference. This can be applied to large files that have no special retention requirements (large system).

Expire After Days Nonusage: 183

Expire Since Days Creation: 731

Primary Storage Days Nonusage: 4

Secondary Storage Days Nonusage: NOLIMIT
Space Management Technique: MIGRATE

STANDRD2

Expiration two years from date of creation and six months from date of last reference. Migrate from primary storage to ML1 one month from the date of last reference, then move the file from ML1 to ML2 after it has been in ML1 for three months. This can be applied to most normal files that have no special retention requirements that can be moved to slower devices.

Expire After Days Nonusage: 183
Expire Since Days Creation: 731
Primary Storage Days Nonusage: 31
Secondary Storage Days Nonusage: 93
Space Management Technique: MIGRATE

BIGFILE

Expiration six months from date of creation and three months from date of last reference. Migrate directly to ML2 seven days from the date of last reference. This can be applied to large files that have no special retention requirements (system migrating to ML1 and ML2).

Expire After Days Nonusage: 93
Expire Since Days Creation: 183
Primary Storage Days Nonusage: 7
Secondary Storage Days Nonusage: 0
Space Management Technique: MIGRATE

HISTORY

No expiration, migrate from primary storage to ML1 20 days from the date of last reference, then move the file from ML1 to ML2 when it has been in ML1 for one year. This management class can be applied to historical files that are not referenced often (system migrating to ML1 and ML2).

Expire After Days Nonusage: NOLIMIT
Expire Since Days Creation: NOLIMIT
Primary Storage Days Nonusage: 20
Secondary Storage Days Nonusage: 366
Space Management Technique: MIGRATE

The ACS Routine

To use the management classes, we need a mechanism to assign management classes to files and directories as they are created, recalled (file only), or converted. The mechanism is the ACS routine. Sample ACS routines have been provided for you, see Appendix D, "Sample ACS Routines," on page 225. The sample ACS routines that were created for the large and small systems are discussed in this section. The sample ACS routine for a large system, SMPACSL ACS, is designed for the use with "the INHERIT value", which is defined in the base configuration file, SMPCNFGL CONFIG. The sample ACS routine for a small system, SMPACSS ACS, is designed for the use with "Invoke ACS for File Creation", which was defined in the base configuration file, SMPCNFGS CONFIG.

Table 8 on page 219 shows some of the variable values set during ACS processing.

Table 8. Values of ACS Read-Only Variables for Sample Files

Object being created	&DIR	&FN	&FT	&PATH	&NDIRS	&PARENTMC
dirID X:IBMUSER	IBMUSER	NULL	NULL	NULL	0	See 1 below.
file N1 T1 in X:IBMUSER	NULL	N1	T1	IBMUSER	1	See 2 below.
dirID X:IBMUSER.DFSMS22	DFSMS22	NULL	NULL	IBMUSER	1	See 2 below.
dirID X:IBMUSER.DFSMS22.DATA	DATA	NULL	NULL	IBMUSER.DFSMS22	2	See 3 below.
file N1 T1 in X:IBMUSER.DFSMS22	NULL	N1	T1	IBMUSER.DFSMS22	2	See 3 below.
file N1 T1 in X:IBMUSER.DFSMS22.DATA	NULL	N1	T1	IBMUSER.DFSMS22.DATA	3	See 4 below.

Table 8. Values of ACS Read-Only Variables for Sample Files (continued)

Object being created	&DIR	&FN	&FT	&PATH	&NDIRS	&PARENTMC
Values for &PARENTMC:						
1. NULL because there is no parent when enrolling a general user.						
2. The management class assigned to directory X:IBMUSER .						
3. The management class assigned to directory X:IBMUSER.DFSMS22 .						
4. The management class assigned to directory X:IBMUSER.DFSMS22.DATA .						

Converting SFS Storage Groups

With a configuration active on each system, management classes need to be assigned to the existing files and directories. Assume you have a file pool (VMSYS1) with four storage groups (2, 3, 4, 5) managed by DFSMS/VM. To convert these storage groups, the following commands are issued by an authorized user.

```
DFSMS CONVERT ST0igroup 2 VMSYS1:
DFSMS CONVERT ST0igroup 3 VMSYS1:
DFSMS CONVERT ST0igroup 4 VMSYS1:
DFSMS CONVERT ST0igroup 5 VMSYS1:
```

The commands can be issued at the same time, or issued individually over several hours or days.

Conversion in the Large System

Table 9 on page 220 shows the resulting management classes for directories and files that were converted with the ACS routine shown in Figure 114 on page 227. When a file or directory is being converted, the directory in which it resides has already been converted. Files being converted in this example are assigned a null value as shown in the ACS sample routine for a large system.

Table 9. Sample Files and Directories for the Large System

File	Directory	Management Class Before	Management Class After
	IBMUSER	none	STANDARD
TEST ASSEMBLE	IBMUSER	none	null
TEST LISTING	IBMUSER	none	null
	EDUC01	none	TEMPFILE
TEST ASSEMBLE	EDUC01	none	null
TEST DUMP	EDUC01	none	null
TEST LISTING	EDUC01	none	null
	IBMUSER.NOEXPIRE	none	STANDARD
TEST ASSEMBLE	IBMUSER.NOEXPIRE	none	null
TEST LISTING	IBMUSER.NOEXPIRE	none	null
	PAYROLL	none	PAYROLL
	PAYROLL.1989DATA	none	PAYROLL
AUG01 SALARY	PAYROLL.1989DATA	none	null
SEP10 SALARY	PAYROLL.1989DATA	none	null
	SQLUSER	none	SQLFILE
DBA SQLDATA	SQLUSER	none	null
Note: If the File field is blank, that indicates a conversion of the directory itself and not a file within the directory.			

Conversion in the Small System

If you take the same files from the previous section and convert them with the ACS routine shown in Figure 112 on page 225, you get the results shown in Table 10 on page 221.

Table 10. Sample Files for the Small Environment

File	Directory	Management Class Before	Management Class After
	IBMUSER	none	TEMPFILE
TEST ASSEMBLE	IBMUSER	none	TEMPFILE
TEST LISTING	IBMUSER	none	TEMPFILE
	IBMUSER.NOEXPIRE	none	TEMPFILE
TEST ASSEMBLE	IBMUSER.NOEXPIRE	none	NOEXPIRE
TEST LISTING	IBMUSER.NOEXPIRE	none	NOEXPIRE
	EDUC01	none	NEWUSER
TEST ASSEMBLE	EDUC01	none	NOEXPIRE
TEST DUMP	EDUC01	none	NULL
TEST LISTING	EDUC01	none	TEMPFILE
	PAYROLL	none	NEWUSER
	PAYROLL.1989DATA	none	null
AUG01 SALARY	PAYROLL.1989DATA	none	null
SEP10 SALARY	PAYROLL.1989DATA	none	null
	SQLUSER	none	NEWUSER
DBA SQLDATA	SQLUSER	none	null

Note: If the File field is blank, that indicates a conversion of the directory itself and not a file within the directory.

Note: During ACS processing for a directory, the path does not include the directory being processed. This is demonstrated in the example above for the directory IBMUSER.NOEXPIRE., where the "Management Class After" is TEMPFILE. The &PATH value for directory IBMUSER.NOEXPIRE is IBMUSER; therefore, the ACS routine for a small system assigns management class TEMPFILE.

Managing a Storage Group

This example displays the results of managing a storage group. Assume that DFSMS/VM has been operating on the system. Now that the file pools being managed by DFSMS/VM have been converted, the storage group is managed. For the examples, assume that all of the files defined in the previous sections reside in the same storage group.

Managing the Large Environment

Assume that the files shown in Table 9 on page 220 reside in storage group 3 of file pool VMSYS1 and the days since creation and last reference of the sample files are shown in Table 11 on page 221.

Table 11. Sample Dates for Files in the Large System

File	Directory	Days Since Creation	Days Since Reference	Management Class Used
TEST ASSEMBLE	IBMUSER	400	21	STANDARD
TEST LISTING	IBMUSER	740	185	STANDARD
TEST DUMP	EDUC01	20	20	TEMPFILE

Table 11. Sample Dates for Files in the Large System (continued)

File	Directory	Days Since Creation	Days Since Reference	Management Class Used
TEST ASSEMBLE	EDUC01	200	5	TEMPFILE
TEST LISTING	EDUC01	5	5	TEMPFILE
TEST ASSEMBLE	IBMUSER.NOEXPIRE	65	21	STANDARD
TEST LISTING	IBMUSER.NOEXPIRE	21	21	STANDARD
AUG01 SALARY	PAYROLL.1989DATA	40	21	PAYROLL
SEP10 SALARY	PAYROLL.1989DATA	0	0	PAYROLL
DBA SQLDATA	SQLUSER	400	2	SQLFILE

Note: The files in this table have a NULL management class as shown in Table 9 on page 220. When a NULL file or directory is managed, the management class of its parent directory is used, as shown in this example.

What happens when the following command is issued:

```
DFSMS MANAGE STORgroup 3 VMSYS1:
```

Assume that the current utilization of storage group 3 is 94.99%. If the STORGROUP_HIGH_THRESHOLD value in the control file (DGTVCNTL DATA) is 95 or higher, the command performs no processing because the storage group is not considered full enough to warrant processing. Now assume that your installation set the STORGROUP_HIGH_THRESHOLD value to 90. Because the current utilization is greater than 90%, this storage group is processed.

First of all, erasure of expired files is performed. The files that match the expiration criteria established by the management class are:

- TEST ASSEMBLE in VMSYS1:EDUC01. (250KB)
- TEST LISTING in VMSYS1:IBMUSER. (350KB)
- TEST DUMP in VMSYS1:EDUC01. (900KB)

This means that TEST ASSEMBLE in the EDUC01 directory is deleted because it was created over seven days ago. TEST LISTING in the IBMUSER directory is deleted because it was created over 2 years ago and has not been referenced in the last six months. TEST DUMP in the EDUC01 is also deleted because it was created over seven days ago.

Now DFSMS/VM looks at the storage group utilization again. Let us assume that the expiration of those three files brought the current utilization to 80.1% of capacity and the low threshold as defined with the STORGROUP_LOW_THRESHOLD value in the control file is 80%. Because the storage group has not reached the low threshold yet, DFSMS/VM attempts to migrate some files. Of the files identified, four are eligible for migration. The four files are:

- TEST ASSEMBLE in VMSYS1:IBMUSER. (20KB)
- TEST ASSEMBLE in VMSYS1:IBMUSER.NOEXPIRE. (12KB)
- TEST LISTING in VMSYS1:IBMUSER.NOEXPIRE. (99KB)
- AUG01 SALARY in VMSYS1:PAYROLL.1989DATA. (100KB)

The algorithm used to migrate files is based on size and age. Because each of these four files has just become eligible for migration, age really is not a factor here. DFSMS/VM migrates AUG01 SALARY to get the biggest gain. If this migration reduces the storage group utilization below 80%, this is the only file migrated. Let us assume that it did not drop below 80%, and the storage group utilization is now 80.01%. DFSMS/VM now migrates TEST LISTING in VMSYS1:IBMUSER.NOEXPIRE and stops because this migration drops utilization below 80%.

The following are the results of using this command: 1500KB are deleted by expiration and 199KB (compacted down to approximately 100KB) are migrated to secondary storage, making 1.7MB available for use by other files.

Managing the Small System

In this system, there is no migration, so the benefit is expiration. Assume that the files shown in [Table 10 on page 221](#) reside in storage group 2 of file pool GLOBAL1 and the days since creation and last reference of these files are shown in [Table 12 on page 223](#).

Table 12. Sample Dates for Files in the Small System

File	Directory	Days Since Creation	Days Since Referenced	Management Class Used
TEST ASSEMBLE	IBMUSER	100	9	TEMPFILE
TEST LISTING	IBMUSER	9	8	TEMPFILE
TEST ASSEMBLE	IBMUSER.NOEXPIRE	100	22	NOEXPIRE
TEST LISTING	IBMUSER.NOEXPIRE	22	8	NOEXPIRE
TEST ASSEMBLE	EDUC01	50	4	NOEXPIRE
TEST DUMP	EDUC01	3	2	NEWUSER
TEST LISTING	EDUC01	4	4	TEMPFILE
AUG01 SALARY	PAYROLL.1989DATA	40	22	NEWUSER
SEP10 SALARY	PAYROLL.1989DATA	0	0	NEWUSER
DBA SQLDATA	SQLUSER	1000	801	NEWUSER

Note: Files with null management classes use the management class of their parent directory. If the parent directory management class is null, then the system default management class will be used. The system default management class is NEWUSER in this example.

Now look at what happens when the following command is issued:

```
DFSMS MANAGE STORgroup 2 GLOBAL1:
```

Comparing the management classes to the dates associated with the files, we see that the following files have expired and are eligible for erasure:

- TEST ASSEMBLE in GLOBAL1:IBMUSER. (200 KB)
- TEST LISTING in GLOBAL1:IBMUSER. (350 KB)
- DBA SQLDATA in GLOBAL1:SQLUSER. (800 KB)

Since this system is not set up for migration, the command completes successfully after freeing up 1.35MB of storage in file pool GLOBAL1 (storage group 2 in particular).

Appendix D. Sample ACS Routines

PI

This appendix contains Programming Interface Information.

Three ACS routines are provided:

- The first routine assigns management classes that have no migration capabilities (see [Figure 112 on page 225](#)).
- The second routine assigns various management classes to files requiring different management classes (see [Figure 114 on page 227](#)). This routine is used with a configuration that has ML1 only (no ML2 capability) and has the INHERIT option.
- The third routine uses the DEFER option for file creation and migrates files to both ML1 and ML2.

Sample ACS Routine Without Migration to ML1

```
Proc 1 mgmtclas

/*****
/* This ACS routine assigns a management class for small systems.      */
/* The RECALL case is not considered since nothing gets MIGRATED.      */
*****/

Filtlist LISTING include(LIST*, 'TEMP') /* Generalized list file types */
Filtlist NOEXP   include(NOEXP*)        /* Classes that don't expire   */

/*****
/* The following filter will match all directories in the IBMUSER      */
/* file space except any ending with NOEXPIRE. This doesn't match when*/
/* a user is enrolled because the path is null at the time.           */
*****/

Filtlist IBMUSER Include(IBMUSER.***) Exclude(IBMUSER.***.NOEXPIRE)

/*****
/* The following select statement only has one branch, but it is      */
/* being included so that the ACS routine can be easily expanded in    */
/* the future.                                                         */
*****/

Select
  When(&acsenvir = 'CREATE' | &acsenvir = 'CONVERT') Do
```

Figure 112. Sample ACS Routine for a Small System (Part 1 of 2)

```

/*****
/* Assign a management class to files or directories while the      */
/* ACS environment is either CREATE or CONVERT.                    */
/*****
/* For newly created or converted files or directories:          */
/* - assign mgmtclas = 'TEMPFILE' for file type of 'TEMP' or file */
/*   type beginning with 'LIST'.                                  */
/* - assign mgmtclas = 'NOEXPIRE' for any file type of 'ASSEMBLE'.*/
/* - assign mgmtclas = 'NOEXPIRE' for any other files or         */
/*   directories with a parent management class beginning        */
/*   with 'NOEXP'.                                                */
/* NOTE: mgmtclas is set to null here. It is unchanged only because */
/*   it was set to null previously.                                */
/*****
Select
  When(&ft = &LISTING)
    Set &mgmtclas = 'TEMPFILE'
  When(&ft = 'ASSEMBLE')
    Set &mgmtclas = 'NOEXPIRE'
  Otherwise
    If &parentmc = &NOEXP Then
      Set &mgmtclas = 'NOEXPIRE'
End
/*****
/* Now that the basic generic processing has been done, the following*/
/* special processing is done to override the generic processing.    */
/*****
/* - If the directory in which the object resides is NOEXPIRE,      */
/*   then assign management class = 'NOEXPIRE'.                    */
/* - If the owner is IBMUSER and the lowest directory is not      */
/*   NOEXPIRE (noexpire's were excluded from the IBMUSER filtlist */
/*   above) assign management class = 'TEMPFILE'.                  */
/* - If this is a new user, assign management class = 'NEWUSER'.   */
/* Write a message in the log file.                                  */
/*****

  If (&path(&ndirs) = 'NOEXPIRE')
    Then
      Set &mgmtclas = 'NOEXPIRE'
  If (&path = &IBUSER | &dir = 'IBUSER')
    Then
      Set &mgmtclas = 'TEMPFILE'
  If (&path = '' && &mgmtclas = '')
    Then Do
      Set &mgmtclas = 'NEWUSER'

      Write 'A new user was just enrolled in the file pool'
    End
  End
End
End
End
/* end do */
/* end when */
/* end select &acsenvir */
/* end proc */

```

Figure 113. Sample ACS Routine for a Small System (Part 2 of 2)

Sample ACS Routine With Migration to ML1

Figure 114 on page 227 shows a method of assigning various management classes to files in an environment requiring many different management classes.

```
Proc 1 mgmtclas

/*****
/* This ACS routine assigns a management class for large systems.  */
*****/

Filtlist NOMIGEX Include(*SQL*)          /* Don't Migrate or Expire */
Filtlist EDUCAT Include(EDUC%%) /*Temporary files owned by Education*/

Select
  When(&acsenvir = 'CREATE' | &acsenvir = 'CONVERT') do
/*****
/* Assign a management class to directories while the ACS */
/* environment is either CREATE or CONVERT. Files will be */
/* assigned a Null value so that they will later be managed by */
/* their parent management classes. */
*****/
/* For newly created or converted directories or files: */
/* - assign mgmtclas = Null if its a file. */
/* - assign mgmtclas = 'TEMPFILE' for any directory with a */
/*   six character Owner beginning with 'EDUC'. */
/* - assign mgmtclas = 'SQLFILE' for any directory with the string */
/*   'SQL' imbedded in its name. */
/* - assign mgmtclas = 'PAYROLL' for a top level directory */
/*   of 'PAYROLL'. */
/* - assign mgmtclas = 'PAYROLL' for directory with a parent */
/*   management class 'PAYROLL' */
/* - assign mgmtclas = 'STANDARD' for any other directories */
/*   that do not meet the prior criteria. */
*****/
    Select
      When(&dir = ' ')
        Set &mgmtclas = ''
      When(&owner = &EDUCAT)
        Set &mgmtclas = 'TEMPFILE'
      When(&dir = &NOMIGEX)
        Set &mgmtclas = 'SQLFILE'
      When(&dir = 'PAYROLL' && &path = '')
        Set &mgmtclas = 'PAYROLL'
      Otherwise
        If (&parentmc = 'PAYROLL') then
          Set &mgmtclas = 'PAYROLL'
        Else
          Set &mgmtclas = 'STANDARD'
    End
  End
End
```

Figure 114. Sample ACS Routine for Large Systems (Part 1 of 2)

```

When(&acsenvir = 'RECALL') do
/*****
/* Assign management classes to any large file being Recalled */
/*****
/*****
/* For very large files (Over 500 kb) */
/* - Do not change mgmtclas for files managed by management class */
/* 'SQLFILE'. */
/* - Do not change mgmtclas for files managed by management class */
/* 'TEMPFILE'. */
/* - Assign mgmtclas 'MIGRFAST' for files managed by management */
/* class 'PAYROLL'. */
/* - Assign mgmtclas = 'LDEFAULT' for files managed by management */
/* class 'STANDARD'. */
/*****
If (&size > 500kb) Then
  Select
    When(&parentmc = 'SQLFILE')
      Set &mgmtclas = ''
    When(&parentmc = 'TEMPFILE')
      Set &mgmtclas = ''

    When(&parentmc = 'PAYROLL')
      Set &mgmtclas = 'MIGRFAST'
    When(&parentmc = 'STANDARD')
      Set &mgmtclas = 'LDEFAULT'

  End
End
/* End select */
/* End when */

End
/* End select on &acsenvir */
/* End Proc */

```

Figure 115. Sample ACS Routine for Large Systems (Part 2 of 2)

Sample ACS Routine With Migration to ML1 and ML2

Figure 116 on page 229 shows a method of using the file creation DEFER option and migration to both ML1 and ML2.

```
Proc 1 mgmtclas

/*****
/* This ACS routine assigns a management class for systems that
/* use the configuration file SMPACS2 CONFIG. This configuration
/* uses the DEFER option for File Create and Migrates files to both
/* ML1 and ML2.
*****/

Filtlist NOMIGEX Include('EXEC','MODULE') /* Don't Migrate or Expire */

Filtlist TEMP Include(EDUC%,'LIST',*TEMP*) Exclude('EDUC99','TEMP99')
/*Temporary files, Don't Migrate */

/* directory names or file names starting with HISTORY */
Filtlist HIST1 Include(HISTORY*)
/* Paths that contain a directory name starting with HISTORY */
Filtlist HIST2 Include(**.HISTORY**.**)

/*****
/* Assign a management class to a directories.
*****/
/*
/* - assign mgmtclas = 'PAYROLL' if the directory name is 'PAYROLL'
/* and the owner is 'PAYDEPT'.
/* - assign mgmtclas = 'PAYROLL' if a directory's parent directory
/* has a management class 'PAYROLL'.
/* - assign mgmtclas = 'HISTORY' if any occurrence of a directory's
/* path or directory name begins with 'HISTORY'.
/* - assign mgmtclas = Null for any directory that does not meet
/* the prior criteria, so that files in it will be managed
/* by the System Default management class; STANDRD2.
*****/

If &dir ~= ''
Then do
Select
When(&dir = 'PAYROLL' && &owner = 'PAYDEPT')
Set &mgmtclas = 'PAYROLL'
When(&parentmc = 'PAYROLL')
Set &mgmtclas = 'PAYROLL'
When(&dir = &HIST1 | &path = &HIST2)
Set &mgmtclas = 'HISTORY'
Otherwise
Set &mgmtclas = ''
End
End
```

Figure 116. Sample ACS Routine with Migration to ML1 and ML2 (Part 1 of 2)

```

/*****
/* Assign a management class to files that have been created with */
/* the DEFER option. */
/*
/* - assign mgmtclas = Null to a file who's parent management class*/
/* is 'PAYROLL', so that the file will be managed by it's */
/* parent directory. */
/* - assign mgmtclas = 'BIGFILE' to migrate fast a file if it is */
/* larger than 500kb. */
/* - assign mgmtclas = 'HISTORY' to move files to slower devices */
/* if the file has a file name beginning with 'HIST'. */
/* - assign mgmtclas = 'TEMPFILE' for any file with a file type */
/* with a six character file type beginning with 'EDUC', */
/* a file type of 'LIST', or a file type with 'TEMP' */
/* embedded. Exclude file types 'EDUC99' and 'TEMP99' */
/* - assign mgmtclas = 'NOEXPIRE' for a file not to be migrated or */
/* expired it is an EXEC or a Module. */
/* - assign mgmtclas = Null for any file that does not meet the */
/* prior criteria, so that it will be managed by it's parent */
/* directory's management class, or if it's parent */
/* directory's management class is Null, the file will be */
/* managed by the System Default management class. */
*****/

If &dir = ''
Then do
Select
When(&parentmc = 'PAYROLL')
Set &mgmtclas = ''
When(&size > 500kb)
Set &mgmtclas = 'BIGFILE'
When(&fn = &HIST1)
Set &mgmtclas = 'HISTORY'
When(&ft = &NOMIGEX)
Set &mgmtclas = 'NOEXPIRE'
When(&ft = &TEMP)
Set &mgmtclas = 'TEMPFILE'
Otherwise
Set &mgmtclas = ''
End
End

End                                     /* End Proc */

```

Figure 117. Sample ACS Routine with Migration to ML1 and ML2 (Part 2 of 2)

Appendix E. Sample EXEC for Automated Migration

Although DFSMS/VM provides no direct support of automatic migration of storage groups, it is possible to set up an EXEC to issue DFSMS MANAGE commands on various days of the week, or at various times of the day. For example, [Figure 118 on page 231](#) shows an EXEC that could be set up to manage all of the DFSMS/VM managed storage groups in a system, with each storage group being processed twice a week.

```
/******  
/* AUTOMIG EXEC  
/*  
/* An EXEC to manage the storage groups of file pools  
/* being managed by DFSMS/VM. The storage groups are divided  
/* into 3 groups, each of which are managed twice during the  
/* week on different days.  
/*  
/* If it's Monday or Thursday morning, manage  
/* storage group 2 with ML2 option in file pool VMSYS1,  
/* storage group 3 with ML1 option in file pool VMSYS1,  
/* and storage group 5 in file pool VMSYS1.  
/*  
/* If it's Tuesday or Friday morning, manage storage group 2  
/* in file pool GLOBAL1. This is a large storage group,  
/* so it is managed by itself.  
/* In this example, (LEVELS ML2 WAIT) is run against storage  
/* group 2 and (LEVELS ML1 NOWAIT) is run against the same  
/* storage group. Since two manages cannot be run against  
/* the same storage group at the same time, the WAIT  
/* demonstrates how to get around this before the EXEC goes on.  
/*  
/* If it's Wednesday or Saturday morning, manage storage group  
/* 3 in file pool GLOBAL1 and storage group 4 in file  
/* pool VMSYS1.  
/*  
/* If it's Sunday night, manage nothing.  
/******  
today = TRANSLATE( DATE(WEEKDAY) )  
Select  
  When (today = 'MONDAY' | today = 'THURSDAY') Then Do  
    'DFSMS MANAGE STORgroup 2 VMSYS1(LEV ML2)'  
    rc1 = rc  
    'DFSMS MANAGE STORgroup 3 VMSYS1(LEV ML1)'  
    rc2 = rc  
    'DFSMS MANAGE STORgroup 5 VMSYS1'  
    rc3 = rc  
    save_rc = Max(rc1,rc2,rc3) /* Save the worst of the three codes */  
  End  
  When (today = 'TUESDAY' | today = 'FRIDAY') Then Do  
    'DFSMS MANAGE STORgroup 2 GLOBAL1(LEV ML2 WAIT)'  
    rc1 = rc  
    'DFSMS MANAGE STORgroup 2 GLOBAL1(LEV ML1 NOWAIT)'  
    rc2 = rc  
    save_rc = Max(rc1,rc2) /* Save the worst of the two codes */  
  End  
  When (today = 'WEDNESDAY' | today = 'SATURDAY') Then Do  
    'DFSMS MANAGE STORgroup 3 GLOBAL1'  
    rc1 = rc  
    'DFSMS MANAGE STORgroup 4 VMSYS1'  
    rc2 = rc  
    save_rc = Max(rc1,rc2) /* Save the worst of the two codes */  
  End  
  Otherwise save_rc = 0 /* Don't do anything on Sunday */  
End  
/******  
/* If it is Saturday morning (Friday night), submit the job to */  
/* run in 2 days (Monday morning), otherwise submit it to run */  
/* tomorrow  
/******  
If today = 'SATURDAY' Then  
  'BATCH SUBMIT AUTOMIG ( CLASS AUTOLOG WBEGIN 0000 WEND OPEN IN 2'  
Else  
  'BATCH SUBMIT AUTOMIG ( CLASS AUTOLOG WBEGIN 0000 WEND OPEN IN 1'  
Exit save_rc /* Exit with RC from DFSMS command */
```

Figure 118. Sample EXEC to Automate Storage Group Management

This EXEC runs in a batch environment with Class AUTOLOG. The IN parameter on the BATCH command causes the job to be resubmitted in the specified number of days.

Alternatively, timers can be set up such that one storage group gets migrated every hour during the night. With REXX and a simple timer program, the installation can set up the migration so it has the least impact on its production systems.

Appendix F. ISMF Variables

PI

This appendix contains Programming Interface Information.

ISMF variables are provide for you to use in your own user extension programs.

You can write your own ISMF extensions to access and extract information from newly created and saved ISMF lists. For information about initiating user extensions, see [Chapter 15, “User Extensions Reference,”](#) on [page 171](#).

Control Variables for ISMF lists

[Table 13](#) on [page 233](#) contains the control variables available for you to perform functions operating on lists displayed on the ISMF list display panels:

- File List panel in the File Application
- Minidisk List panel in the Minidisk Application
- Management Class List panel in the Management Class Application
- Saved ISMF Lists panel in the Saved Lists Application

Table 13. Common Control Variables for the ISMF lists			
Variable Name	Description	Attribute	Length
DGTCMSG	Saves short message text	character	35
DGTCLMSG	Saves long message text	character	79
DGTMSGID	Contains a message ID if one is supplied	character	8
DGTERRRW	Error entry	binary	31
DGTCUROW	Current row pointer	binary	31
DGTRCODE	Return code	binary	31
DGTCONTN	Directs the list processor to either continue or stop processing the list depending on the return code. You can set this variable to one of the following values: A blank directs the processor to continue processing the list entries until they have all been operated against or an error occurs. This is the default. N Directs the processor to stop processing the list entries. Y Directs the processor to continue processing the list entries regardless of the return code. Processing continues until all list entries have been operated against, but list entries causing the error are not processed.	character	1

Table 13. Common Control Variables for the ISMF lists (continued)

Variable Name	Description	Attribute	Length
DGTLISTC	Controls the line operator history for the current list entry. You can set this variable to one of the following values: Y Updates the line operator history for the current list entry. This is the default. N Does not update the line operator history for the current list entry.	character	1
DGTCMDNM	Contains the name of the current list command	character	8
DGTCOUNT	Contains the number of the list entry being operated against	binary	31
DGTLASTU	Indicates whether last-use or normal mode was requested. You can set this variable to one of the following values: Y last-use mode N normal mode	character	1
DGTTOTAL	Contains the total number of entries in the list (excluding hidden entries or entries that have been filtered out).	character	6
DGTTYPEC	Indicates the type of command in control. You can set this variable to one of the following values: C indicates that the command in control was invoked as a list command. L indicates that the command in control was invoked as a line operator.	character	1

File List Panel Column Variables

Table 14 on page 235 lists available variables for use in any user extensions written for operations against File List panel entries.

Table 14. Variables for a File List Entry

Column	Variable Name	Attribute	Length
Display flag	CDISPFLG	character	1
Filter flag	CFILTFLG	character	1
Hide flag	CHIDEFLG	character	1
Line operator	CLINEOP	character	10
File entry name	COBJ	character	28
Allocated space	CFLALSPC	character	10
Number of records	CFLNUMRC	character	10
Record length	CFLRECLN	character	10
Create date	CFLCREDT	character	10
Reference date	CFLREFDT	character	10
Management class name	CFLMGCNM	character	8
Management class origin	CFLMCASN	character	6
File owner	CFLFLOWN	character	8
Directory owner	CFLDROWN	character	8
Authority	CFLAUTHR	character	9
Migrate status	CFLMIGFL	character	3
File name	CFLFILNM	character	8
File type	CFLFILTP	character	8
File mode	CFLFILMD	character	2
Record format	CFLRECFM	character	3
Parent directory	CFLDIRID	character	153
Short message	CSMSG	character	35
Long message	CLMSG	character	79
Type	CFLTYPE	character	8

Minidisk List Panel Column Variables

Table 15 on page 236 lists available column variables for use in any user extensions written for operations against Minidisk List panel entries.

Table 15. Column Variables for Minidisk List Panel

Column	Variable Name	Attribute	Length
Display flag	CDISPFLG	character	1
Filter flag	CFILTFLG	character	1
Hide flag	CHIDEFLG	character	1
Line operator	CLINEOP	character	10
Minidisk list entry	CBJEC	character	14
Minidisk start	CDSTART	character	10
Minidisk end	CDEND	character	10
Minidisk size	CDSIZE	character	10
Minidisk gap	CDGAP	character	10
Device volser	CEVVOLSR	character	6
Device type	CEVTYPE	character	2
Duplex status	CPLXSTAT	character	8
Fast write status	CDFWSTAT	character	8
Read cache status	CDCASTAT	character	8
Cache fast write status	CAFWSTAT	character	8
System ID	CYSTEMID	character	8
Minidisk address	CDBADDR	binary	4
Minidisk start	CDBSTART	binary	4
Minidisk end	CDBEND	binary	4
Request ID	CEQUESTID	binary	4
Short message	CSMSG	character	35
Long message	CLMSG	character	79
Minidisk owner	COBJ	character	8
Minidisk address	CDADDR	character	4

Management Class List Panel Column Variables

Table 16 on page 237 lists available variables for use in any user extensions written for operations against Management Class List panel entries.

Table 16. Variables for a Management Class List Entry

Column	Variable Name	Attribute	Length
Display flag	CDISPFLG	character	1
Filter flag	CFILTFLG	character	1
Hide flag	CHIDEFLG	character	1
Line operator	CLINEOP	character	10
Management class name	COBJ	character	8
Expire nonusage	CEXPIRNU	character	7
Expire since creation	CEXPIRDD	character	7
Expiration disposition	CEXPDISP	character	8
Primary storage non-usage days	CPRIMDAY	character	7
Space management technique	CMIGRATE	character	7
Last modified user ID	CLUSERID	character	8
Last modified date	CLDATE	character	10
Last modified time	CLTIME	character	8
Configuration file name and type	CCFNT	character	17
Configuration file mode	CCFFM	character	1
Configuration file directory ID	CCDIR	character	153
Short message	CSMSG	character	35
Long message	CLMSG	character	79
Secondary storage non-usage days	CSECSTOR	character	7

Saved ISMF Lists Panel Column Variables

Table 17 on page 238 lists available column variables for use in any user extensions written for operations against Saved ISMF Lists panel entries.

Table 17. Column Variables for the Saved ISMF Lists Panel

Column	Variable Name	Attribute	Length
Display flag	CDISPFLG	character	1
Filter flag	CFILTFLG	character	1
Hide flag	CHIDEFLG	character	1
Line operator	CLINEOP	character	10
List name	COBJ	character	8
List type	CLLTYPE	character	8
List date	CLU4DATE	character	10
List time	CLUTIME	character	8
List user	CLLUSER	character	8
List row count	CLLRCNT	character	8
List update	CLLUPD	character	8
Short message	CSMSG	character	35
Long message	CLMSG	character	79
(Not Displayed)	CLUDATE	character	8

Column Variables for Saved Lists

Table 18 on page 239 contains the conversion rule for obtaining the column names of the saved lists. The only difference between the column name for a newly generated list and the name of the corresponding column in the saved list is that the first character is different. The actual value of this character depends on the application.

For example, there is a column name CFLMIGFL listed in file application as listed in Table 14 on page 235. The first character for file application is F as listed in Table 18 on page 239. From these two, the name of the corresponding column in the saved file list is FFLMIGFL. The name of any saved list column in any application can be deduced in a similar manner.

Table 18. Conversion Table for Column Names

Type of list	First Character
Saved File List	F
Saved Management Class List	R
Saved Minidisk List	Y

Appendix G. Sample User Extension Program — PRNTMINI

PI

This appendix contains Programming Interface Information.

This appendix consists of:

- A listing of PRNTMINI (used to print a list of minidisks), that is a sample user extension program included in the DFSMS/VM Function Level 221 product tape
- A sample file tailoring skeleton used with PRNTMINI
- Sample output from the PRNTMINI EXEC

PRNTMINI EXEC

```
/*
/* *****
/*
/* REXX: PRNTMINI
/*
/* (C) COPYRIGHT IBM CORPORATION 1989, 1993.
/*
/* DESCRIPTIVE NAME: ISMF MINIDISK LIST PRINT
/*
/* STATUS: DFSMS/VM FUNCTION LEVEL 221
/*
/* FUNCTION:
/* THIS EXEC OPENS A SAVED ISMF MINIDISK LIST AND FORMATS IT
/* USING THE ISPF FILE TAILORING SKELETON DGTKPRMD
/*
/* INPUT:
/* INNAME TEST
/* ISPSLIB MEMBER DGTKPRMD
/* ISPSLIB MEMBER BLANK
/*
/* OUTPUT: ISPF FILE MEMBER OUTNAME
/* DGTRCODE
/*
/* PROCESSOR: ISPF DIALOG MANAGER FILE TAILORING SERVICES
/*
/* CHANGE ACTIVITY:
/* $V4=VMLISTAP,HDF1125,89/08/28,STLMRD: VM/ISMF SAVED LIST APPLIC.
/* $P1=KAR0014 ,HDF1125,89/10/13,STLMRD: Correct missing comma @P1A*
/* $P2=KVE0134 ,HDF2210,90/06/04,STLCKD: Correct spelling error @P2A*
/* $S2=KGD0471,2.2.1,920929,SJPLVK. Errors in EXEC @S2C*
/* *****
/* PARSE UPPER ARG INNAME TEST REST;
/* IF INNAME ^= ' ' | TEST ^= ' ' | REST ^= ' ' THEN
/* SAY 'PRNTMINI PARAMETERS: ' INNAME TEST REST;
/* Call Init;
/* ADDRESS ISPEXEC ' VGET (ZAPPLID) SHARED';
/* *****
/* ENSURE THAT THE APPLICATION ID IS DGT2, ISMF LIST APPLICATION
/* *****
/* IF ZAPPLID ^= 'DGT2' THEN
/* DO
/* IF SUBSTR(ZAPPLID,1,3) ^= 'DGT' THEN /* @P1C*/
/* DO /* NOT ISMF */
/* SAY 'PRNTMINI FAILED - MUST BE EXECUTED FROM ISMF';
/* DGTRCODE = 20;
/* Return DGTRCODE;
/* END /* NOT ISMF */
/* ELSE
/* DO
/* SAY 'PRNTMINI FAILED - MUST BE USED IN ISMF LIST APPLICATION';
/* DGTCONTN = 'N';
/* DGTLISTC = 'N';
```

```

        DGTRCODE = 20;
        Call Set_exit;
        Return DGTRCODE;
    END
END
/*****
/*
/* Obtain the command name, total and count values from the ISPF
/* shared pool where ISMF has placed them
/*
/*
*****/
ADDRESS ISPEXEC ' CONTROL ERRORS RETURN';
ADDRESS ISPEXEC ' VGET ,
        ( DGTCOUNT DGTTOTAL DGTCMDNM ZDATE ZTIME)  SHARED';
COMMAND = DGTCMDNM
/*****
/*
/* Obtain the list panel column values for the current entry in
/* a similar way. Then, if TEST mode, display each column. Do
/* not actually execute the PRNTMINI function, if TEST mode.
/*
/*
*****/
ADDRESS ISPEXEC 'VGET ( CDISPFLG CFILTFLG CHIDEFLG CLINEOP COBJ,
        CLLTYPE CLUDATE CLUTIME CLLUSER CLLRCNT
        CLLUPD CMSG CLMSG ) SHARED'; /*@S2C*/
/*@S2C*/
IF RC = 0 THEN
    DO;
        DGTCLMSG= 'PRNTMINI OF 'INNAME' FAILED: (VGET R/C:'RC')';
        DGTCSMSG= 'PRNTMINI FAILED';
        DGTRCODE = 4 ;
        Call Set_exit;
        Return DGTRCODE;
    END;
Call Init_vars;
IF TEST = 'TEST' THEN
    DO ;
        Call Display_Test;
        DGTRCODE = 4;
        Call Set_exit;
        Return DGTRCODE;
    END
ELSE
    DO ;
        /*****
        /* BEGIN EXEC MAINLINE
        *****/
        IF CLLTYPE = 'MINIDISK' THEN
            DO ;
                /* NOT A MINIDISK LIST, SKIP IT */
                DGTCLMSG = 'INNAME' IS NOT A MINIDISK LIST, IT WAS NOT PROCESSED'
                DGTCSMSG = 'NOT A MINIDISK LIST';
                DGTRCODE = 4;
                CALL SET_EXIT;
                RETURN DGTRCODE;
            END
        /*****
        /* Open the table (ISMF saved list)
        *****/
        Address ISPEXEC 'TBOPEN 'INNAME' NOWRITE';
        IF RC = 0 THEN
            DO;
                DGTCLMSG= 'PRNTMINI OF 'INNAME' FAILED: (TBOPEN R/C:'RC')';
                DGTCSMSG= 'PRNTMINI FAILED';
                DGTRCODE = 8 ;
                Call Set_exit;
                Return DGTRCODE;
            END;
        ELSE
            DO;
                OPENTB = '1';
            END;
        /*****
        /* Open the file tailoring file
        *****/
        Address ISPEXEC 'FTOPEN ' ;
        IF RC = 0 THEN
            DO;
                DGTCLMSG= 'PRNTMINI OF 'INNAME' FAILED: (FTOPEN R/C: 'RC')';
                DGTCSMSG= 'PRNTMINI FAILED';
                DGTRCODE = 20;
                Call Set_exit;
                Return DGTRCODE;
            END;
        ELSE
            DO;
                DGTCLMSG= 'PRNTMINI OF 'INNAME' FAILED: (FTOPEN R/C: 'RC')';
                DGTCSMSG= 'PRNTMINI FAILED';
                DGTRCODE = 20;
                Call Set_exit;
                Return DGTRCODE;
            END;
        END;
    END;

```

```

        END;
    ELSE
        DO ;
            OPENFT = '1';
        END;
    /******
    /* Do the FTINCL to create the output member in the file
    /* associated with the FILEDEF $DGTFILE
    /******
    address ISPEXEC 'FTINCL 'SKEL ;
    IF RC = 0 THEN
        DO
            DGTCLMSG= 'PRNTMINI OF 'INNAME' FAILED: (FTINCL 'SKEL' R/C: 'RC')';
            DGTCMSG= 'PRNTMINI FAILED';
            DGTRCODE = 20;
            Call Set_exit;
            Return DGTRCODE;
        END
    ELSE
        DO;
            FTINCL = '1';
        END ;
    /******
    /* END OF PRNTMINI
    /******
    DGTCLMSG = 'INNAME' PROCESSED AT 'ZTIME' ON 'ZDATE' BY PRNTMINI';
    DGTCMSG = 'PRNTMINI COMPLETED ';
    END;
    CALL SET_EXIT;
    RETURN DGTRCODE;
End;

SET_EXIT:
/******
/* SET_EXIT: CLEAN UP AND SET RETURN CODE AND MSGS BEFORE EXIT */
/******

/******
/* CLOSE THE FILE TAILORING FILE IF NEEDED
/******
IF OPENFT = '1' THEN
    DO;
        IF DGTRCODE < 8 THEN
            DO;
                ADDRESS ISPEXEC 'FTCLOSE NAME('OUTNAME')';
                IF RC = 0 THEN
                    DO;
                        DGTCLMSG= 'PRNTMINI OF 'INNAME' FAILED: (FTCLOSE R/C: 'RC')';
                        DGTCMSG= 'PRNTMINI FAILED';
                        DGTRCODE = MAX (DGTRCODE,8);
                    END;
                END;
            ELSE
                DO;
                    Address ISPEXEC 'FTCLOSE';
                    IF RC = 0 THEN
                        DO;
                            DGTCLMSG= 'PRNTMINI OF 'INNAME' FAILED: (FTCLOSE R/C: 'RC')';
                            DGTCMSG= 'PRNTMINI FAILED';
                            RC = MAX (DGTRCODE,8);
                            DGTRCODE = RC;
                        END;
                    END;
                END;
            END;
        END;
    /******
    /* CLOSE THE SAVED LIST IF NEEDED
    /******
    IF OPENTB = '1' THEN
        DO;
            address ISPEXEC 'TBCLOSE 'INNAME ;
            IF RC = 0 THEN
                DO;
                    DGTCLMSG= 'PRNTMINI OF 'INNAME' FAILED: (TBCLOSE R/C: 'RC')';
                    DGTCMSG= 'PRNTMINI FAILED';
                    RC = MAX (DGTRCODE,8);
                    DGTRCODE = RC;
                END;
            END;
        END;
    /******
    /* Set RETURN CODE, MSGS and FLAGS
    /******

```

```

/*****
ADDRESS ISPEXEC 'VPUT ( DGTRCODE DGTCMSG DGTCLMSG ,
                      DGTCONTN DGTLISTC ,
                      ) SHARED ' ;

      IF RC /= 0 THEN                                     /*@S2C*/
      DO;
        SAY          'PRNTMINI OF 'INNAME' FAILED: (VPUT R/C: 'RC')';
        DGTRCODE = 20;
      END;
RETURN DGTRCODE;

Display_test:
/*****
/* Display_test: Show the column values from the shared pool */
/*****
SAY
SAY 'LIST ENTRY 'DGTTCOUNT' OF 'DGTTOTAL', HAS THE FOLLOWING VALUES:'
SAY ' '
SAY '          COL      VARIABLE      CURRENT'
SAY '          TAG      NAME          VALUE'
SAY '          ---      - - - - -      - - - - -'
SAY '          (--)      CDISPFLG:      'CDISPFLG'
SAY '          (--)      CFILTFLG:      'CFILTFLG'
SAY '          (--)      CHIDEFLG:      'CHIDEFLG'
SAY '          (1 )      CLINEOP        'CLINEOP'
SAY '          (2 )      COBJ           'COBJ'
SAY '          (3 )      CLLTYPE        'CLLTYPE'
SAY '          (4 )      CLUDATE        'CLUDATE'
SAY '          (5 )      CLUTIME        'CLUTIME'
SAY '          (6 )      CLLUSER        'CLLUSER'
SAY '          (7 )      CLLRCNT        'CLLRCNT'
SAY '          (8 )      CLLUPD         'CLLUPD'
SAY '          (--)      CMSG           'CMSG'
SAY '          (--)      CLMSG          'CLMSG'
SAY '          (--)      DGTTCOUNT      'DGTTCOUNT'
SAY '          (--)      DGTTOTAL       'DGTTOTAL'
SAY '          (--)      DGTCONTN      'DGTCONTN'
SAY ' '
      DGTCLMSG = INNAME' WAS PROCESSED IN TEST MODE, IT HAS BEEN SKIPPED';
      DGTCMSG = 'PRNTMINI IN TEST MODE';
Return 0;

Init:
/*****
/* Init: initialize the EXEC */
/*****
DGTRCODE = 0;
DGTCMSG = '';
DGTCLMSG = '';
DGTCONTN = '';                                     /*@S2A*/
DGTLISTC = 'Y';
OPENTB = 0
OPENFT = 0
Return 0;

Init_vars:
/*****
/* Init_vars: set variables */
/*****
INNAME = COBJ;
OUTNAME = INNAME;
SKEL = 'DGTKPRMD';
/*****
/* Complete the header used by the skeleton */
/*****
U20DUDAT = CLUDATE;
U20DUTIM = CLUTIME;
U20DUSER = CLLUSER;
Return 0;

```

SAMPLE FILE TAILORING SKELETON

```
)CM *****
)CM **
)CM SKELETON: DGTKPRMD **
)CM **
)CM (C) COPYRIGHT IBM CORPORATION 1989, 1990. **
)CM **
)CM DATE OF LAST CHANGE: 10/04/05 **
)CM **
)CM DESCRIPTIVE NAME: ISMF SAVED LIST APPLICATION: FORMAT A SAVED **
)CM MINIDISK LIST **
)CM **
)CM STATUS: DFSMS/VM FUNCTION LEVEL 221 **
)CM **
)CM FUNCTION: **
)CM **
)CM THIS IS A FILE TAILORING SKELETON WHICH CREATES A **
)CM LIST WITH: **
)CM (1) A HEADER BOX THAT IDENTIFIES THE LIST, WHEN IT WAS **
)CM FORMATTED AND BY WHOM; WHEN IT WAS LAST UPDATED AND **
)CM BY WHOM **
)CM (2) A SET OF LIST COLUMN HEADINGS, LIKE THE ONES THAT **
)CM ISMF USES WHEN DISPLAYING THE LIST AT THE TERMINAL **
)CM (3) THE MINIDISK LIST DISPLAYED UNDER COLUMN HEADINGS **
)CM (4) A TRAILER BOX THAT DENOTES THE END OF THE LIST AND **
)CM STATES THE NUMBER OF LIST ITEMS **
)CM **
)CM PROCESSOR: ISPF **
)CM **
)CM CHANGE ACTIVITY: **
)CM **
)CM *****
)CM *****
)CM CLEAR THE COUNTERS
)CM *****
)SET TLCNT = 0
)CM *****
)CM SET TABS FOR FORMATTING (EXCLAMATION POINT)
)CM *****
)TB 14 26 72
)CM *****
)CM GENERATE THE HEADINGS LINE
)CM *****
)-----?
--
-- ISMF MINIDISK LIST: &INNAME!--
--! !!--
)TB 72
)TB 14 26 72
-- PRINTED:!!&TDATE!AT: &ZTIME HRS!--
-- BY:!!&ZUSER!!--
--! !!--
-- UPDATED:!!&U20DUDAT!AT: &U20DUTIM HRS!--
-- BY:!!&U20DUSER!!--
)-----?
--
)CM *
)CM *****
)CM GENERATE AN OUTPUT LINE FOR EACH TABLE ENTRY
)CM *****
)CM *
)CM *****
)CM SET TABS FOR FORMATTING (EXCLAMATION POINT)
)CM *****
)TB 2 13 22 28 39 50 61 72 79 87 96 105 114
!LINE !MDISK !MDISK! MDISK ! MDISK ! MDISK ! MDISK !?
DEVICE!DEVICE ! DUPLEX !DASD FW !RD CACHE!CACHE FW
!OPERATOR !OWNERID !ADDR ! START ! END ! SIZE ! GAP !?
VOLSER!TYPE ! STATUS ! STATUS ! STATUS ! STATUS
!---(1)---!---(2)---!---(3)---!---(4)---!---(5)---!---(6)---!---(7)---!?
--(8)---!---(9)---!---(10)---!---(11)---!---(12)---!---(13)---
)DOT &INNAME
)SET TLCNT = &TLCNT + 1
!&YLINEOP!&YOBJ!&YDADDR!&YDSTART!&YDEND!&YDSIZE!&YDGAP!&YEVVOLSR!&YEVVTY?
PE!&YPLXSTAT!&YDFWSTAT!&YDCASTAT!&YAFWSTAT
```

```

)ENDDOT
)CM *****
)CM SET TABS FOR FORMATTING (EXCLAMATION POINT)
)CM *****
)TB      72
)CM *****
)CM GENERATE THE END OF LIST LINE
)CM *****
)BLANK
-----?
-----
--                      END OF LIST!--
)SEL &TLCNT EQ 0
--  THERE ARE NO MINIDISKS LISTED!--
)ENDSEL
)SEL &TLCNT NE 0
--  MINIDISKS LISTED-&TLCNT!--
)ENDSEL
--!--
-----?
-----

```

Sample PRNTMINI Output

```

-----
--              ISMF MINIDISK LIST: SAMPLE              --
--
-- PRINTED: 2005/04/10 AT: 10:50 HRS.                  --
-- BY: SMITH                                           --
--
-- UPDATED: 2005/04/10 AT: 13.01.29 HRS.              --
-- BY: GEORGE                                          --
-----
LINE   MDISK   MDISK   MDISK   MDISK   MDISK   MDISK   DEVICE  DEVICE  DUPLEX  DASD FW  RD CACHE  CACHE FW
OPERATOR OWNERID  ADDR   START   END     SIZE    GAP    VOLSER  TYPE    STATUS  STATUS   STATUS    STATUS
---(1)--- --(2)--- --(3)--- --(4)--- --(5)--- --(6)--- --(7)--- --(8)--- --(9)--- --(10)--- --(11)--- --(12)--- --(13)---
PHIL    127      0       0       49      50      0    ASTRIX  9345    DUPLEX  ACTIVE   ACTIVE    ACTIVE
ANANTHA A07       0       0       0       1      0    HYGENX  3380    NONE    NONE    INACTIVE  NONE
KATHY   191      1       5       5       5      0    DGMATX  3390    DUPLEX  ACTIVE   INACTIVE  INACTIVE
DAVE    193      6       8       3       3      0    GETAFX  3390    PENDING  NONE    ACTIVE    ACTIVE
CAROL   195      9      11      3       3      0    HYGENX  3380    NONE    NONE    INACTIVE  NONE
DICK    196     12     14      3       3      0    DGMATX  3390    DUPLEX  ACTIVE   INACTIVE  INACTIVE
MATT    1CA     15     18      4       4      0    GETAFX  3390    PENDING  NONE    ACTIVE    ACTIVE
GREG    1A5     19     21      3       3      0    HYGENX  3380    NONE    NONE    INACTIVE  NONE
BOB     1A0     22     24      3       3      0    ASTRIX  9345    DUPLEX  ACTIVE   ACTIVE    ACTIVE
CRAIG   112     25     26      2       2      0    OBELIX  3380    SIMPLEX  INACTIVE  INACTIVE  INACTIVE
DAN     39D     27     36     10      10      0    DGMATX  3390    DUPLEX  ACTIVE   INACTIVE  INACTIVE
DON     2C2     37     49     13      13      0    HYGENX  3380    NONE    NONE    INACTIVE  NONE
LUZ     12D      0     799    800      5      0    GETAFX  3390    DUPLEX  ACTIVE   INACTIVE  INACTIVE
PAULA   5C2      1      5      5       5      0    OBELIX  3380    SIMPLEX  INACTIVE  INACTIVE  INACTIVE
JIM     5C4      6     10      5       5      0    OBELIX  3380    SIMPLEX  INACTIVE  INACTIVE  INACTIVE
-----
--                      END OF LIST                      --
-- MINIDISKS LISTED-15                                   --
--
-----

```

Appendix H. Implementing Configurations

PI

This appendix contains Programming Interface Information.

To make DFSMS/VM work for you, you need to create, use, and manipulate configurations. This section covers some of the basic steps in creating configurations and some hints on configuration use and manipulation to make your job easier. This is intended to supplement the contents of the following chapters:

- Chapter 5, “Configurations,” on page 53
- Chapter 13, “Managing Storage Using DFSMS/VM,” on page 117
- Chapter 6, “Automatic Class Selection,” on page 63

How do I use a configuration to automatically assign management classes?

1. Create your configuration and ACS processing logic.
 - Create your configuration to contain base configuration information and management classes.
 - Create your ACS processing logic in one of the following three ways:
 - If you use an ACS routine, then write the routine and translate it into the configuration. For additional information, see [Chapter 16, “ACS Language Reference,” on page 177](#).
 - If you use a REXX exit, write the exit, compile it (compiling is highly recommended), and place it in IGDACSMC DFSMS in the SFS directory VMSYS:DFSMS.ACSEXITS. For additional information, see [Appendix A, “ACS REXX Exit,” on page 191](#).
 - If you use a module exit, write, assemble, load and genmod the exit and place the load module in IGDACSMC MODULE in the SFS directory VMSYS:DFSMS.ACSEXITS. For additional information, see [Appendix B, “ACS Module Exit,” on page 203](#).
2. Activate your configuration using ISMF (see “Activating a Configuration” on page 59) or the DFSMS ACTIVATE command (see “DFSMS ACTIVATE” on page 135). After successful activation, the configuration and ACS processing is used by DFSMS/VM.

What are the contents of the configuration?

- Base configuration information—The name of the default management class, the invoke ACS for file creation indicator, and a description. The description should contain enough detail so you can easily identify the contents. You can use the ISMF Configuration Application to view the ACTIVE configuration, and if the configuration description is well defined you can easily identify which configuration is currently active.
- Management classes—The names given to management classes should indicate the type of data they are to control, rather than the attributes of the management class. By using this type of naming convention, only the management class attributes need to change if the data handling requirements change.

For example, if payroll data is assigned a management class of PAYROLL, and after evaluation of use you decide to migrate the data a week sooner than you are currently migrating, only alter the management class attributes of PAYROLL and activate the altered configuration. If your management class name reflects how the data is handled (for example, MIGAT6WK), you would need to assign a new management class (MIGAT5WK) to all the payroll data.

- ACS routine—The name of the ACS source file translated into the configuration is stored in the configuration. Therefore, a meaningful ACS source file name helps to identify the function of a configuration. For example:

```
STANDARD ACS1292 for the ACS source routine written for your normal
processing as of December 1992.
```

Note: The ACS routine is optional if you use an installation-wide exit.

How can I make a new configuration?

- Using ISMF, perform the following steps in any order, but it is helpful to perform these steps in the order in which they are listed.
 - Use the Management Class Application to create management classes. You must define at least the management class named as default when the base configuration is defined, but you also want to create all the management classes you will need to manage your system (see [“How can I modify configurations?”](#) on page 250 for more information).
 - Use the Configuration Application to create the base configuration information.
 - Use the Automatic Class Selection Application to translate your ACS routine into a configuration. You may create or edit your ACS source routine outside of ISMF, but you must translate it any time the ACS routine is changed.
- Use the CMS COPY command to copy an existing configuration into a new configuration, and use ISMF to modify the new configuration. You may want to do this if you wish to create a new configuration that is similar to an existing configuration.

If you make any changes to your configuration, you must activate it for the changes to become effective. If you change the ACS routine, you must translate the ACS routine, then activate the configuration for the changes to become effective. Also you should run DFSMS CONVERT with the REDETERMINE option to insure all files and directories have the correct management class.

How can I modify configurations?

- Automatic Class Selection Application—The existing ACS routine is replaced or removed, or an ACS routine is added to the configuration using this application. If you want to change from using an ACS routine to using a REXX exit, use the ACS Application to delete the ACS object (the translated routine). If you want to add or replace an ACS routine, translate the new routine into this configuration.
- Management Class Application—Existing management classes are deleted or changed, or new management classes are added using this application.

Use the Management Class List application to see what management classes already exist. You can delete, copy, or alter management classes from this list. Copy is useful for creating new management classes with similar attributes; the copy panel allows you to make an exact copy, or enter the alter panels to make changes. Copy can also be used to copy management classes from one configuration to another.

- Configuration Application
 - Base configuration information—Modify the description to allow you to identify the altered configuration. Change the default management class or the Invoke ACS for File Creation indicator, or both, if you want these to be different.

If you make any changes to your configuration, you must activate it for the changes to become effective. If you change the ACS routine, you must translate the ACS routine, then activate the configuration for the changes to become effective. Also to insure that all files and directories have the correct management class, run DFSMS CONVERT with the REDETERMINE option.

What can I use to automatically assign management classes?

- ACS routine becomes part of the source configuration when it is translated
- REXX installation-wide exit
- Module installation-wide exit

Note: The installation-wide exits are associated with a configuration at the time of activation. Since the exits must have a specified name to be used, keep a separate copy of the source with a file name and file type that helps you differentiate them and identify which configuration you intend to use them with.

Because no verification is done to ensure that you do not activate an exit that assigns invalid management classes, it is very important that you only use exits together with the configuration for which they are designed. For example, if your configuration source file ID is NORMAL CNFG0193, your REXX source file ID might be NORMAL REXX0193.

How do I put the exits, ACS routine, and configuration together and use them?

Example: GLOBALFP:SYSADMIN. contains

```
NORMAL  CNFG0193  - source configuration file
NORMAL  ACS0193   - source ACS routine
NORMAL  REXX0193  - source REXX exit
NORMAL  ASSM0193  - source assembler exit
```

To use the above files:

- NORMAL ACS0193 is translated into NORMAL CNFG0193 (a configuration that contains the management classes assigned by NORMAL ACS0193, NORMAL REXX0193, and NORMAL ASSM0193).
- NORMAL REXX0193 is copied into IGDACSMC DFSMS in the SFS directory VMSYS:DFSMS.ACSEXITS, or it is compiled and the output from IBM compiler for REXX/370 is placed in IGDACSMC DFSMS in the SFS directory VMSYS:DFSMS.ACSEXITS.
- NORMAL ASSM0193 is assembled and the module file is created, and placed in IGDACSMC MODULE in the SFS directory VMSYS:DFSMS.ACSEXITS.

After the translation is completed and the exits are placed in the appropriate files, NORMAL CNFG0193 is activated. All four source files are kept for reference, and if changes are needed, new source files would be created.

Note: It is not recommended that you have more than one item performing ACS processing. The previous information is only an example.

How do I stop using an ACS routine?

To stop using an ACS routine, delete the ACS routine from the source configuration (using the ACS Application delete function) and activate the configuration, or create a new source configuration without an ACS routine and activate the new source configuration.

Note: You should provide either an installation-wide REXX exit or module exit to handle ACS processing.

How do I stop using a module or REXX exit?

To stop using a module or REXX exit, erase the file in SFS directory VMSYS:DFSMS.ACSEXITS (IGDACSMC DFSMS (REXX) or IGDACSMC MODULE (module)) and activate a configuration.

Note: You need to provide an ACS routine or other type of exit to handle ACS processing.

How do I use configurations and ACS to manage files at a directory level?

To manage files at a directory level, files must have the NULL management class. A configuration that has the "invoke ACS for file creation" field set to INHERIT produces files that have the NULL management class. A configuration that has the "invoke ACS for file creation" set to DEFER, however, produces files with no management class. Also files created before DFSMS/VM is installed or while DFSMS/VM is down have no management class. To ensure all files have the NULL management class, the ACS routine or exits must assign the NULL management class to files during convert and recall processing. Files created while a previous configuration was active can have a management class other than the NULL management class.

If files with a management class exist on the system, a DFSMS CONVERT command with the REDETERMINE option must be run to convert all files to the NULL management class. Files with no management class are converted during MANAGE or MIGRATE command processing if a CONVERT command has not been processed.

Note: You can also choose to only use directory-level management on selected files. In this case, you have to ensure that the ACS routine or exit assigns the NULL management class to these files in all environments (CREATE, CONVERT and RECALL).

How do I assign the NULL management class?

- With a configuration—Files created while the active configuration has "invoke ACS for file creation" set to INHERIT, are assigned the NULL management class. This is effective only for files created while DFSMS/VM is running with this configuration and that have not been recalled. For other files ACS processing must be used to assign the NULL management class.
- With ACS processing—Files are assigned the NULL management class during create, recall, and convert processing; and directories are assigned the NULL management class during convert and create processing. This is accomplished:
 - By default—Only file and directories without a management class are assigned the NULL management class by default. There are two ways to cause this:
 - The ACS routine or exits do not assign a management class for the file or directory being processed. This is not recommended.
 - No ACS routines or exits are associated with the active configuration.
 - By explicit assignment—In the ACS routine the statement SET &MGMTCLAS = '' is used. For additional information, see [Appendix A, "ACS REXX Exit,"](#) on page 191 and [Appendix B, "ACS Module Exit,"](#) on page 203.

Note: Default assignment of the NULL management class using ACS routine or exits is not recommended. Explicit assignment allows the ACS routine or exit to clearly show which management classes are used, and avoids the problem shown in the following ACS routine examples:

```
PROC MGMTCLAS
/* Assign BLUE to directories containing the word blue, GREEN to */
/* directories containing the word green, and OTHER to all other */
/* directories. */
FILTLIST BLUE INCLUDE(*BLUE*)
FILTLIST GREEN INCLUDE(*GREEN*)
SELECT(&DIR)
  WHEN (&BLUE)
    SET &MGMTCLAS = 'BLUE'
  WHEN (&GREEN)
    SET &MGMTCLAS = 'GREEN'
  OTHERWISE
    SET &MGMTCLAS = 'OTHER'
END
END
```

The intent of this routine is to assign the management class *OTHER* to all directories that do not have *BLUE* or *GREEN* as part of the name and to assign the NULL management class to the default files. Instead it also assigns the management class *OTHER* to all files. If the routine is coded in the following manner, the NULL management class will actually be assigned:

```
PROC MGMTCLAS
/* Assign BLUE to directories containing the word blue, GREEN to */
/* directories containing the word green, and OTHER to all other */
/* directories. Assign the NULL management class to files. */
FILTLIST BLUE INCLUDE(*BLUE*)
FILTLIST GREEN INCLUDE(*GREEN*)
SELECT(&DIR)
  WHEN (')
    SET &MGMTCLAS = ' ' /* for file processing directory */
    /* variable is null */
  WHEN (&BLUE)
    SET &MGMTCLAS = 'BLUE'
  WHEN (&GREEN)
    SET &MGMTCLAS = 'GREEN'
  OTHERWISE
    SET &MGMTCLAS = 'OTHER'
END
END
```

How do I change management classes?

- Changing management class attributes—If you are only changing the attributes of management classes (for example; changing the number of days it takes for a file with this attribute to expire), do the following:

1. Alter the management classes in the source configuration
2. Activate the changed configuration

All files with the management classes that were changed are now managed by the new attribute. The DFSMS CONVERT command does not need to be run, since the management class name is already associated with the files.

- Adding or deleting management classes—If you are adding or deleting management classes, do the following:

1. Change configuration and ACS routines and exits.
 - Change the source configuration, adding or deleting the management classes or both.
 - Change the ACS routine and exits to use the new management classes, and to assign a management class from the updated configuration to all files and directories that have a management class that has been deleted.
2. If you are using an ACS routine, translate it into your changed configuration. If you are using ACS exits, follow the instructions in the exit appendixes for exit activation.
3. Activate the changed configuration.
4. If you deleted management classes, or created new management classes and want existing files to use them, issue the CONVERT command with the REDETERMINE option. This needs to be done on all storage groups in all file pools on the system.

Remember: If you are sharing configurations across multiple z/VM systems and are using ACS exits, you must change the exits in the VMSYS file pool on each system before activating the configuration on those systems.

Appendix I. Multi-Language Support

DFSMS/VM is available in three languages:

- Mixed-case English (AMENG)
- Upper-case English (UCENG)
- Japanese (KANJI)

In addition to supporting these languages, DFSMS/VM supports multiple active languages. For example, a general user can have a virtual machine set to upper-case English, issue a DFSMS MIGRATE command, and the returned reader file is in upper-case English. In a similar manner, another general user could have a virtual machine set to Japanese, issue a DFSMS MIGRATE command (at the same time the other MIGRATE is issued), and the returned reader file would be in Japanese. This requires that the language features for the desired language be installed for the base CP and CMS products, and that the required DFSMS/VM language features are specified. Specification of the language feature can only take place after installation of DFSMS/VM. See the *Program Directory for function level 221* for language feature installation instructions.

There is, however, a performance penalty for language switching. This penalty is incurred in the DFSMS master and server machines where the messages are actually created. The penalty is greater when informational messages are logged to the DFSMS logs in one language, but DFSMS/VM is processing many commands that return report files in a language different from the language that the DFSMS logs use.

Note: ISMF fails when the system language is set to Japanese and the terminal does not support Japanese.

Appendix J. DFSMS/VM Accounting Services

DFSMS/VM produces two types of accounting records:

- User record—DFSMS/VM account record type 'USMS'
- System record—DFSMS/VM account record type 'SSMS'

User records are produced as the result of processing in ISMF, file or directory processing, or issuing a DFSMS command. The user record contains information that can be charged to the specific action that has generated the record. The system records are produced as the result of DFSMS/VM overhead that cannot be charged to a specific action; for example, DFSMS/VM initialization, termination, and logging.

Accounting records are 80-character card images, by z/VM convention. They are produced by the DFSMS/VM masters or servers via Diagnose X'4C' with subcode X'0010'.

To use the DFSMS/VM accounting function, you must setup the system virtual machines for CP accounting. For additional information about CP accounting, refer to the *z/VM: CP Planning and Administration*. To produce DFSMS/VM accounting records, set the accounting option in the DFSMS/VM control file to **Y** and specify the ACCT operand in the option control statement of the z/VM CP directory entries for the DFSMS/VM master and servers.

An accounting exit provides an installation the opportunity either to modify an accounting record just prior to DFSMS/VM generating the record or to stop a record from being generated. Accounting services provide a modifiable data buffer to the accounting exit. The buffer contains data from columns 9–78 of either the user or system accounting records.

All DFSMS/VM accounting records have record type CO in columns 79–80 of the account record. This is generated by z/VM and indicates that the record was produced with Diagnose X'4C'.

Note: Refer to the *z/VM: DFSMS/VM Customization* for additional information about accounting services and the accounting exit.

USER Accounting Record

Table 19 on page 257 provides a format description of the DFSMS/VM USER accounting record type.

Table 19. Column Description of the USER Accounting Record

Column	User Record Contents
1–8	User ID of the DFSMS/VM virtual machine, set by CP, char(8).
9–16	User ID of the virtual machine for which resources are being reported (command issuer), char(8).
17–32	Installation supplied data (for example, account numbers), char(16). This is represented by binary zeros unless replaced by installation supplied data.
33–46	Date and time of the accounting record (yyyymmddhhmmss), char(14).
47–50	Command being processed, char(4), see “User Accounting Record Command Codes” on page 258.
51–54	Transaction identifier associated with this request, binary fullword.
55–58	Milliseconds of elapsed time ³ used while processing this request, binary fullword.
59–62	Milliseconds of CPU processor time used processing this request, binary fullword.
63–66	Number of 4K blocks written to ML1 for this request, binary fullword.
67–70	Number of 4K blocks written to ML2 for this request, binary fullword.

Table 19. Column Description of the USER Accounting Record (continued)

Column	User Record Contents
71–74	Number of 4K blocks expired, written during recall, or number of cylinders moved to the target minidisk for this request, binary fullword.
75–78	DFSMS/VM account record type 'USMS' ('xSMS', where x is 'U' for USER).
79–80	Accounting record identification code ('C0'), set by CP.

Note: Fields referring to the number of 4K blocks include data blocks only.

User Accounting Record Command Codes

The command field (columns 47–50) in a User accounting record contains the DFSMS/VM process generating the accounting record. These processes include DFSMS commands, DFSMSRM commands, ISMF requests, and internal processes associated with DFSMS and SFS (erasure of migrated data, automatic recall of a file, ACS processing for file creation...). The command field can be used to selectively generate accounting records.

The command can be one of the following:

ACTV

Activate from ISMF or the DFSMS command

ALTR

Alter from ISMF or the DFSMS command

ARCL

Automatic recall generated by SFS

BLDL

Minidisk build list

CHCK

Minidisk check

CNS

Configuration and management class requests from ISMF

CONV

Convert from the DFSMS command

DELE

Delete from DFSMS command

DISC

Discard from ISMF or the DFSMS command

DTMC

Determine management class (ACS processing) (generated by SFS)

ERAS

Erasure of migrated data (generated by SFS)

MANG

Manage from the DFSMS command

MIGR

Migrate from ISMF or DFSMS command

MOVE

Minidisk move

³ The elapsed processing time is not as accurate as CPU utilization. Elapsed time can be less than CPU utilization for extremely short running DFSMS/VM processes.

QRYR	Query request from ISMF or DFSMS command
QRYs	Query status from DFSMS command
QRYL	Query minidisk list from DFSMS command
RECL	Recall from ISMF or DFSMS command
REPF	Report file space from DFSMS command
REPS	Report storage group from DFSMS command
STOP	Stop from DFSMS command
TEST	Test ACS from ISMF ACTIVATE command
RDIS	Discard from DFSMSRM command
RDMT	Demount from DFSMSRM command
RMNT	Mount from DFSMSRM command
RQRY	Query request from DFSMSRM command
RQRL	Query library from DFSMSRM command
RSTP	Stop from DFSMSRM command
RDCT	Set/Reset DEVCAT from DFSMSRM command
RVCT	Set VOLCAT from DFSMSRM command

System Accounting Record

Table 20 on page 259 shows the format of the DFSMS/VM system accounting record type of 'SSMS'.

<i>Table 20. System Accounting Record Column Description</i>	
Column	SYSTEM Record Contents
1–8	User ID of the DFSMS/VM virtual machine, set by CP, char(8).
9–16	'DFSMSsYS' representing DFSMS system, char(8).
17–32	Installation supplied data (for example, account numbers), char(16). This is represented by zeros unless replaced by installation-supplied data.
33–46	Date and time of the accounting record (yyyymmddhhmmss), char(14).
47–58	Reserved.
59–62	Milliseconds of CPU processor time used by DFSMS/VM.
63–66	Number of 4K blocks logged by DFSMS/VM, binary fullword.

Table 20. System Accounting Record Column Description (continued)

Column	SYSTEM Record Contents
67–74	Reserved.
75–78	DFSMS/VM account record type 'SSMS' ('xSMS', where x is 'S' for SYSTEM).
79–80	Accounting record identification code ('C0'), set by CP.

Appendix K. Year 2000 Enhancements for DFSMS/VM

APAR VM61214 provides year 2000 support and enhancements for DFSMS/VM in the following areas:

- Fixed window usage has been standardized to 1950-2049.
- The List Date column on the Saved ISMF Lists Panel has been extended to 10-characters.
- DFSMS COPY and MOVE commands will set the century bit in the minidisk label accordingly.
- Several ISMF help messages have been extended for 4-digit years.
- Several reports have been enhanced to include 4-digit years.
- Corrected problems occurring around the roll over to 2000/01/01.
- Removed the ISMF restriction on the first character of a directory name must be alphabetic.

Fixed Window Date Usage

DFSMS/VM Function Level 221 uses dates quite extensively. It uses DIAGNOSE X'0C' to obtain the current system date then converts it to 10-character through a 1950-2049 fixed window. This means that all 2-digit years that are between 50 and 99 inclusive will be interpreted as 1950-1999. All 2-digit years that are between 00 and 49 inclusive will be interpreted as 2000-2049. This is applied when:

- displaying a date to a panel
- migrating or recalling files
- determining file expiration
- sorting based on date in panels
- filtering
- 10-character dates contained within reports.

Warning!

If running the space management portion of DFSMS/VM, the SFS filepool servers must be at least VM/ESA Version 2 Release 2.0 (or higher). The DFSMS/VM master and space management servers can be on an older release of VM/ESA. The problem is that a pre-year 2000 level of SFS does not update the last reference date past 12/31/99.

Saved ISMF Lists

The DFSMS/VM ISMF application 'Saved ISMF Lists Panel' (panel ID DGTLLA1) was displaying 8-character dates in the 'List Date' field. User written extensions can obtain the value of this field through the CLUDATE ISPF variable.

The List Date field has been extended to display 10-character dates in the form *yyyy/mm/dd*. This date is contained within a new ISPF variable called CLU4DATE. The CLUDATE variable is still available as 8-characters.

[Table 17 on page 238](#) has been enhanced to include the CLU4DATE variable. [Table 17 on page 238](#) lists available column variables for use in any user extensions written for operations against Saved ISMF Lists panel entries.

The Saved Lists Filter Entry Panel (panel ID DGTDLF11), which is reached by issuing the filter command without any operands on the Saved ISMF List Panel, has been enhanced to accept 10-character dates in the List Date Value fields. The filter panel will still accept 8-character dates as input, converting them to 10-character dates using the 1950-2049 fixed window.

The following additional changes have been made for the Saved ISMF List panel:

- Help files for the Saved Lists Filter Entry Panel have been updated to show that a 10-character value can be entered for the List Date fields.
- Message DGTUV008, displayed when a date provided with the filter command on the Saved ISMF Lists Panel or File List Panel is not 8- or 10-characters long or when the date specified is not within the 1950-2049 fixed window, has been enhanced as follows:

Short Message: INVALID DATE

Long Message: Date must be of the form YY/MM/DD or YYYY/MM/DD

Explanation: The date you specified is not in the correct format. The date must be in the form YY/MM/DD or YYYY/MM/DD, where YY is the 2-digit year or YYYY is the 4-digit year, MM is the numeric month of the year, and DD is the numeric day of the month. The date cannot be before the year 1950 or after the year 2049.

- The sample PRNTMINI exec, used on the Saved ISMF Lists Panel to obtain information regarding a saved minidisk list, has been enhanced to use the CLU4DATE in the generated ISMFLIST file. In addition, the PRINTED field contained in the header portion of the ISMFLIST file is now 10-characters in the form *yyyy/mm/dd*. Figure 119 on page 262 shows an example of the extended date fields.

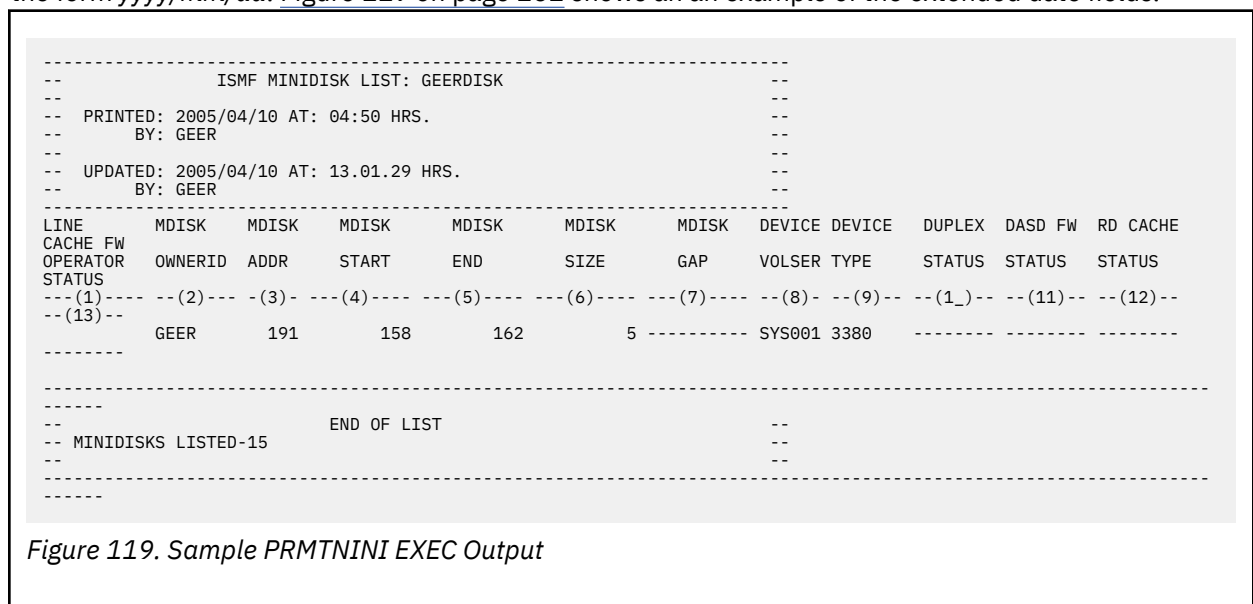


Figure 119. Sample PRMTNINI EXEC Output

- The explanation portion of messages DGTUV011, DGTUV095, and DGTUV096 have been enhanced to indicate the date field can be entered in the form YY/MM/DD or YYYY/MM/DD.

DFSMS COPY and MOVE commands

When using the DFSMS MOVE command to move data from one minidisk to another, there is an option to keep the original create date for the minidisk. These commands have been enhanced to properly set the century bit field within the CMS minidisk label control block. The century bit is used by VM/ESA Version 2 Release 2.0 (or higher), and ignored on older releases.

DFSMS/VM Generated Reports

Date in some reports, but not all, were updated from 8-character to 10-character.

DFSMS CHECK Command Report

The date created and last changed fields of the report generated by the DFSMS CHECK command have been extended to 10-character dates in the form *yyyy/mm/dd*. Figure 120 on page 263 shows an example of the extended date fields.

```

Userid: DFSMS      Date: 04/09/05   Time: 22:55:46
CHECK  191 ( FILE TYPE FOR GEER   AS 191
Address: 0191 Device Type: 3380 Date Created: 1998/01/15 07:29:36
Valid: DFS191 Block size: 4096 Last Changed: 2005/04/09 22:55:33
Cyls:    40 Usable Cyls:    40

Total number of CMS blocks:      6000
Number of CMS blocks used:      1915 ( 32%)
CMS blocks counted:              1915
Blocks in allocation map:        1915

Lost CMS blocks:                  0
Invalid CMS blocks:               0
Overlapping CMS blocks:           0

Disk origin pointer:              5
Files reported in directory:      187 (Including DIRECTOR and ALLOCMAP)
Number of files found:            187

Command CHECK gave return code = 0

```

Figure 120. Sample DFSMS CHECK report

For examples of the DFSMS CHECK command report, see [Figure 64 on page 110](#) and [Figure 78 on page 140](#).

DFSMS REPORT Command Report

The date migrated and date last referenced fields of the report generated by the DFSMS REPORT command have been extended to 10-character dates in the form *mm/dd/yyyy*. [Figure 121 on page 263](#) shows an example of the extended date fields.

```

DFSMS Function Level 221                      01/05/00 05:35:09
DFSMS REPORT SPACEMANAGEMENT FILESPACE, request identifier 114

DFSMS REPORT SPACEMANAGEMENT FILESPACE processing started for file pool DFSMS1 and userid TEST

Files migrated for this file space:
  Migration Level 1:                      102      123
  Migration Level 2:                      21
Logical 4K blocks currently in use by this file space:      143
Physical 4K blocks in use by this file space:               141
Physical 4K blocks in use by primary storage for this file space: 20
Physical 4K blocks in use by secondary storage for this file space: 121
  Migration Level 1:                      100
  Migration Level 2:                      21
Physical 4K blocks saved for this file space due to compaction: 2

The following files are migrated:

Filename Filetype Primary 4K Blocks Management Class Date Last Referenced Date Migrated Migration Level
Directory DFSMS1:TEST.
TEST TEST 1 MEDIUM 01/02/2000 01/05/2000 2
TEST2 TEST 1 MEDIUM 01/03/2000 01/05/2000 2
:

The following files are migrated but DFSMS has migrated data in both ML1 and ML2:

Filename Filetype Primary 4K Blocks Management Class Date Last Referenced Date Migrated Migration Level Token for Delete Command
Directory DFSMS1:TEST.
TEST3 TEST 1 LONG 01/03/2000 01/05/2000 1&2; 00030000000001DD
TEST4 TEST 1 LONG 01/03/2000 01/05/2000 1&2; 00030000000001D9

The following files are migrated but no migrated data exists for them:

Filename Filetype Primary 4K Blocks Management Class Date Last Referenced
Directory DFSMS1:TEST.
TEST EXEC 1 LONG 01/02/2000
TEST FILE 1 LONG 01/02/2000

DFSMS REPORT SPACEMANAGEMENT FILESPACE completed with no errors

```

Figure 121. Sample DFSMS REPORT report

For examples of the DFSMS REPORT command report, see [Figure 69 on page 124](#), [Figure 73 on page 126](#), and [Figure 89 on page 161](#).

Removal of ISMF Alphabetic Restriction on Directory Names

The ISMF component restricted the first character of a directory name to be alphabetic only, it could not be numeric. This restriction has been removed. The following ISMF help messages have been enhanced:

- Message DGTVM037 issued when any character of a directory name is detected as not valid input on the ACS Test Case Define Panel or the ACS Test Case Alter Panel:

Short Message: INVALID DIRECTORY

Long Message: Must be 1 to 16 alphanumeric characters

Explanation: You entered an invalid directory. The directory id can be from 1 to 16 characters. Each character can be alphabetic or numeric.

- The explanation portion of messages DGTVM016, DGTVM017, DGTVM024, DGTVM036, DGTVM042, DGTVM043, and DGTVM045 have been enhanced to indicate a directory qualifier can be from 1 to 16 characters and each qualifier character can be alphabetic or numeric.

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

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

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

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

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

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

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

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

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

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information may contain sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Programming Interface Information

This book primarily documents information that is NOT intended to be used as Programming Interfaces of z/VM.

This book also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of z/VM. This information is identified where it occurs by an introductory statement to a chapter or section.

Trademarks

IBM, the IBM logo, and [ibm.com](https://www.ibm.com/legal/copytrade)® are trademarks or registered trademarks of International Business Machines Corp., in the United States and/or other countries. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on [IBM Copyright and trademark information](https://www.ibm.com/legal/copytrade) (<https://www.ibm.com/legal/copytrade>).

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Adobe, Acrobat, PostScript and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

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

Terms and Conditions for Product Documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal Use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial Use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see:

- The section entitled **IBM Websites** at [IBM Privacy Statement](https://www.ibm.com/privacy) (<https://www.ibm.com/privacy>)
- [Cookies and Similar Technologies](https://www.ibm.com/privacy#Cookies_and_Similar_Technologies) (https://www.ibm.com/privacy#Cookies_and_Similar_Technologies)

Bibliography

This topic lists the publications in the z/VM library. For abstracts of the z/VM publications, see [z/VM: General Information](#).

Where to Get z/VM Information

The current z/VM product documentation is available in [IBM Documentation - z/VM \(https://www.ibm.com/docs/en/zvm\)](https://www.ibm.com/docs/en/zvm).

z/VM Base Library

Overview

- [z/VM: License Information](#), GI13-4377
- [z/VM: General Information](#), GC24-6286

Installation, Migration, and Service

- [z/VM: Installation Guide](#), GC24-6292
- [z/VM: Migration Guide](#), GC24-6294
- [z/VM: Service Guide](#), GC24-6325
- [z/VM: VMSES/E Introduction and Reference](#), GC24-6336

Planning and Administration

- [z/VM: CMS File Pool Planning, Administration, and Operation](#), SC24-6261
- [z/VM: CMS Planning and Administration](#), SC24-6264
- [z/VM: Connectivity](#), SC24-6267
- [z/VM: CP Planning and Administration](#), SC24-6271
- [z/VM: Getting Started with Linux on IBM Z](#), SC24-6287
- [z/VM: Group Control System](#), SC24-6289
- [z/VM: I/O Configuration](#), SC24-6291
- [z/VM: Running Guest Operating Systems](#), SC24-6321
- [z/VM: Saved Segments Planning and Administration](#), SC24-6322
- [z/VM: Secure Configuration Guide](#), SC24-6323

Customization and Tuning

- [z/VM: CP Exit Customization](#), SC24-6269
- [z/VM: Performance](#), SC24-6301

Operation and Use

- [z/VM: CMS Commands and Utilities Reference](#), SC24-6260
- [z/VM: CMS Primer](#), SC24-6265
- [z/VM: CMS User's Guide](#), SC24-6266
- [z/VM: CP Commands and Utilities Reference](#), SC24-6268

- [z/VM: System Operation](#), SC24-6326
- [z/VM: Virtual Machine Operation](#), SC24-6334
- [z/VM: XEDIT Commands and Macros Reference](#), SC24-6337
- [z/VM: XEDIT User's Guide](#), SC24-6338

Application Programming

- [z/VM: CMS Application Development Guide](#), SC24-6256
- [z/VM: CMS Application Development Guide for Assembler](#), SC24-6257
- [z/VM: CMS Application Multitasking](#), SC24-6258
- [z/VM: CMS Callable Services Reference](#), SC24-6259
- [z/VM: CMS Macros and Functions Reference](#), SC24-6262
- [z/VM: CMS Pipelines User's Guide and Reference](#), SC24-6252
- [z/VM: CP Programming Services](#), SC24-6272
- [z/VM: CPI Communications User's Guide](#), SC24-6273
- [z/VM: ESA/XC Principles of Operation](#), SC24-6285
- [z/VM: Language Environment User's Guide](#), SC24-6293
- [z/VM: OpenExtensions Advanced Application Programming Tools](#), SC24-6295
- [z/VM: OpenExtensions Callable Services Reference](#), SC24-6296
- [z/VM: OpenExtensions Commands Reference](#), SC24-6297
- [z/VM: OpenExtensions POSIX Conformance Document](#), GC24-6298
- [z/VM: OpenExtensions User's Guide](#), SC24-6299
- [z/VM: Program Management Binder for CMS](#), SC24-6304
- [z/VM: Reusable Server Kernel Programmer's Guide and Reference](#), SC24-6313
- [z/VM: REXX/VM Reference](#), SC24-6314
- [z/VM: REXX/VM User's Guide](#), SC24-6315
- [z/VM: Systems Management Application Programming](#), SC24-6327
- [z/VM: z/Architecture Extended Configuration \(z/XC\) Principles of Operation](#), SC27-4940

Diagnosis

- [z/VM: CMS and REXX/VM Messages and Codes](#), GC24-6255
- [z/VM: CP Messages and Codes](#), GC24-6270
- [z/VM: Diagnosis Guide](#), GC24-6280
- [z/VM: Dump Viewing Facility](#), GC24-6284
- [z/VM: Other Components Messages and Codes](#), GC24-6300
- [z/VM: VM Dump Tool](#), GC24-6335

z/VM Facilities and Features

Data Facility Storage Management Subsystem for z/VM

- [z/VM: DFSMS/VM Customization](#), SC24-6274
- [z/VM: DFSMS/VM Diagnosis Guide](#), GC24-6275
- [z/VM: DFSMS/VM Messages and Codes](#), GC24-6276
- [z/VM: DFSMS/VM Planning Guide](#), SC24-6277

- *z/VM: DFSMS/VM Removable Media Services*, SC24-6278
- *z/VM: DFSMS/VM Storage Administration*, SC24-6279

Directory Maintenance Facility for z/VM

- *z/VM: Directory Maintenance Facility Commands Reference*, SC24-6281
- *z/VM: Directory Maintenance Facility Messages*, GC24-6282
- *z/VM: Directory Maintenance Facility Tailoring and Administration Guide*, SC24-6283

Open Systems Adapter

- Open Systems Adapter/Support Facility on the Hardware Management Console (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/SC14-7580-02.pdf), SC14-7580
- Open Systems Adapter-Express ICC 3215 Support (<https://www.ibm.com/docs/en/zos/2.3.0?topic=osa-icc-3215-support>), SA23-2247
- Open Systems Adapter Integrated Console Controller User's Guide (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/SC27-9003-02.pdf), SC27-9003
- Open Systems Adapter-Express Customer's Guide and Reference (https://www.ibm.com/docs/en/SSLTBW_2.3.0/pdf/iaa2z1f0.pdf), SA22-7935

Performance Toolkit for z/VM

- *z/VM: Performance Toolkit Guide*, SC24-6302
- *z/VM: Performance Toolkit Reference*, SC24-6303

The following publications contain sections that provide information about z/VM Performance Data Pump, which is licensed with Performance Toolkit for z/VM.

- *z/VM: Performance*, SC24-6301. See *z/VM Performance Data Pump*.
- *z/VM: Other Components Messages and Codes*, GC24-6300. See *Data Pump Messages*.

RACF® Security Server for z/VM

- *z/VM: RACF Security Server Auditor's Guide*, SC24-6305
- *z/VM: RACF Security Server Command Language Reference*, SC24-6306
- *z/VM: RACF Security Server Diagnosis Guide*, GC24-6307
- *z/VM: RACF Security Server General User's Guide*, SC24-6308
- *z/VM: RACF Security Server Macros and Interfaces*, SC24-6309
- *z/VM: RACF Security Server Messages and Codes*, GC24-6310
- *z/VM: RACF Security Server Security Administrator's Guide*, SC24-6311
- *z/VM: RACF Security Server System Programmer's Guide*, SC24-6312
- *z/VM: Security Server RACROUTE Macro Reference*, SC24-6324

Remote Spooling Communications Subsystem Networking for z/VM

- *z/VM: RSCS Networking Diagnosis*, GC24-6316
- *z/VM: RSCS Networking Exit Customization*, SC24-6317
- *z/VM: RSCS Networking Messages and Codes*, GC24-6318
- *z/VM: RSCS Networking Operation and Use*, SC24-6319
- *z/VM: RSCS Networking Planning and Configuration*, SC24-6320

TCP/IP for z/VM

- [*z/VM: TCP/IP Diagnosis Guide*](#), GC24-6328
- [*z/VM: TCP/IP LDAP Administration Guide*](#), SC24-6329
- [*z/VM: TCP/IP Messages and Codes*](#), GC24-6330
- [*z/VM: TCP/IP Planning and Customization*](#), SC24-6331
- [*z/VM: TCP/IP Programmer's Reference*](#), SC24-6332
- [*z/VM: TCP/IP User's Guide*](#), SC24-6333

Prerequisite Products

Device Support Facilities

- Device Support Facilities (ICKDSF): User's Guide and Reference (https://www.ibm.com/docs/en/SSLTBW_2.5.0/pdf/ickug00_v2r5.pdf), GC35-0033

Related Products

XL C++ for z/VM

- [*XL C/C++ for z/VM: Runtime Library Reference*](#), SC09-7624
- [*XL C/C++ for z/VM: User's Guide*](#), SC09-7625

z/OS

IBM Documentation - z/OS (<https://www.ibm.com/docs/en/zos>)

Index

Special Characters

/ (slash character) [172](#)

A

accessing a saved list
 through EXECs [173](#)
 through the minidisk application [92](#)
accessing a saved minidisk list [92](#)
ACS
 application [66](#)
 description of [63](#)
 directory-level indexing function [183](#)
 language constants [177](#)
 language read-only variables [179](#)
 language read/write variables [178](#)
 language reference
 comparison operators [182](#)
 description of [177](#)
 language statements
 DO [187](#)
 END [190](#)
 EXIT [189](#)
 FILTLIST [185](#)
 IF [187](#)
 PROC [184](#)
 SELECT [188](#)
 SET [186](#)
 WRITE [189](#)
 module exit
 (IGDACERO), input variables [208](#)
 (IGDACERW), values [213](#)
 (IGDACSPM), parameter list [207](#)
 errors [207](#)
 object deletion from a source configuration file [73](#)
 object information, display [72](#)
 processing [8](#), [63](#)
 processing steps [65](#)
 REXX exit [191](#)
 routines
 boolean expressions [183](#)
 comparison operators [182](#)
 description of [177](#)
 editing [67](#)
 in a sample environment [219](#)
 invoking [213](#)
 masks [177](#)
 return and reason codes from [215](#)
 test (ACS) [71](#)
 translating [67](#)
 tasks, using ISMF for [66](#)
 test case define or ACS test case alter [69](#)
 test listing [71](#)
 test selection panel [68](#)
 translator listing [68](#)
 variables [192](#), [194](#)

ACSPAERO [213](#)
ACSPAERW [213](#)
ACTIVATE command
 description of [135](#)
 syntax [135](#)
ACTIVATE list command
 description of [163](#)
activating a configuration [59](#)
activation of the module exit [204](#)
active configurations on systems that share data [10](#)
active configurations, initialization [10](#)
ALTER command
 description of [136](#)
 syntax [136](#)
 usage notes [137](#)
ALTER FILESPACE list command
 description of [163](#)
ALTER line operator
 description of [166](#)
altering the base configuration [56](#)
applications
 ACS [66](#)
 configuration [53](#)
 list [30](#)
assigning management classes [42](#)
assignments, setting PF key [35](#)
asterisk (*)
 * asterisk [169](#)
 in line operator history [169](#)
attributes, management class expiration [39](#)
attributes, management class migration [40](#)
attributes, summary of management class [41](#)
authorization and security controls [113](#)
authorization file, DFSMS/VM [113](#)
authorization process for DFSMS commands [114](#)
automated move process [98](#)
automatic recall [121](#)

B

backup
 of primary storage [81](#)
 of secondary storage [81](#)
 using a non-IBM backup product
 [84](#)
backup and recovery processing [81](#)
base configuration define or alter [56](#)
base configuration information [55](#)
boolean expressions
 ACS routine [183](#)
BOTTOM list command
 description of [163](#)
BROWSE line operator
 description of [166](#)

C

- cancelling
 - current task [21](#)
- changing
 - PF key settings [36](#)
 - selection criteria [89](#)
- CHECK command
 - description of [138](#)
 - syntax [138](#)
- CHECK line operator
 - description of [106](#), [165](#)
 - using [107](#)
- check request
 - issuing from ISMF [107](#)
 - results of [106](#)
 - sample output [110](#)
 - sample results [109](#)
- class, management [7](#)
- classes, rules and usage of management [7](#)
- CLEAR list command
 - description of [163](#)
- CMS minidisks to new DASD, moving [93](#)
- codes, reason [207](#)
- codes, return [206](#)
- column variables
 - file list panel [235](#)
 - for saved lists [239](#)
 - management class list panel [237](#)
 - minidisk list panel [236](#)
- command and line operator usage [162](#)
- commands, DFSMS/VM
 - ACTIVATE [135](#)
 - ALTER [136](#)
 - authorization process [114](#)
 - CHECK [138](#)
 - CONVERT [140](#)
 - COPY [141](#)
 - DELETE [144](#)
 - DISCARD [146](#)
 - help [133](#)
 - ISMF [162](#)
 - MANAGE [147](#)
 - MIGRATE [150](#)
 - MOVE [151](#)
 - operational [132](#)
 - processed in the user machine [131](#)
 - processing overview [131](#)
 - QUERY [157](#)
 - RECALL [158](#)
 - recall by [122](#)
 - REPORT [160](#)
 - STOP [161](#)
 - that produce reports [131](#)
 - usage [134](#)
- comparison operators [182](#)
- comparison rules [183](#)
- compiler, using the IBM REXX/370 [191](#)
- configuration
 - activation [59](#)
 - active [10](#)
 - application [53](#)
 - application selection panel [54](#)
 - base [53](#)

- configuration (*continued*)
 - base display [55](#)
 - define or alter [56](#)
 - initialization [10](#)
 - on systems that share data, different active [10](#)
 - valid and invalid [10](#)
 - validation [58](#)
- configurations, DFSMS/VM [9](#), [53](#)
- control file name changes [76](#)
- control file, DFSMS/VM [11](#)
- control to DFSMS/VM, returning [197](#)
- control variables for ISMF lists [233](#)
- conversion in the large system [220](#)
- conversion in the small system [221](#)
- CONVERT command
 - description of [140](#)
 - syntax [140](#)
- converting a storage group [123](#)
- converting SFS storage groups, sample environment [220](#)
- COPY command
 - description of [141](#)
 - restrictions [143](#)
 - syntax [142](#)
- COPY line operator
 - description of [166](#)
- CP/CMS list command
 - description of [163](#)
- CURSOR list command
 - description of [163](#)
- customization considerations [93](#)

D

- DASD, moving CMS minidisks to new [93](#)
- define management class [48](#)
- defining
 - PF keys [34](#)
- defining a MACLIB [174](#)
- defining the base configuration [56](#)
- DELETE command
 - description of [144](#)
 - syntax [144](#)
- DELETE line operator
 - description of [166](#)
- delete, management class [46](#)
- deleting an ACS object from a source configuration file [73](#)
- DFSMS/VM
 - authorization file [113](#)
 - commands
 - ACTIVATE [135](#)
 - ALTER [136](#)
 - authorization process [114](#)
 - CHECK [138](#)
 - CONVERT [140](#)
 - COPY [141](#)
 - DELETE [144](#)
 - DISCARD [146](#)
 - MANAGE [147](#)
 - MIGRATE [150](#)
 - MOVE [151](#)
 - online HELP [133](#)
 - QUERY [157](#)
 - RECALL [158](#)
 - REPORT [160](#)

- DFSMS/VM (*continued*)
 - commands (*continued*)
 - STOP [161](#)
 - usage [134](#)
 - configurations [9](#), [53](#)
 - control file [11](#)
 - customizing [93](#)
 - function level [221](#) [7](#)
 - interface [11](#)
 - managing storage [117](#)
 - processing, starting [111](#)
 - processing, stopping [111](#)
 - returning control to [197](#), [206](#)
 - shared file system [11](#)
 - starting [111](#)
 - stopping [111](#)
 - stopping server virtual machines [111](#)
 - stopping the master virtual machine [111](#)
 - storage, managing [117](#)
 - subcommand environment [195](#)
 - subcommands
 - EXTRACT [195](#)
 - SET [196](#)
 - WRITE [197](#)
- directory path mask examples [178](#)
- directory path mask rules [178](#)
- directory-level indexing function, ACS [183](#)
- DISCARD command
 - description of [146](#)
 - syntax [146](#)
- DISCARD line operator
 - description of [165](#)
- DISCARD list command
 - description of [163](#)
- DISPLAY line operator
 - description of [166](#)
- displaying
 - PF key settings [33](#)
- DO statement (ACS)
 - description of [184](#), [187](#)
- DOWN list command
 - description of [163](#)

E

- edit ACS routines [67](#)
- EDIT line operator
 - description of [166](#)
- END list command
 - description of [21](#), [163](#)
- END statement (ACS)
 - description of [184](#), [190](#)
- entry panel, management class sort [46](#)
- entry parameters [204](#)
- ERASE line operator
 - description of [165](#)
- errors, ACS module exit [207](#)
- errors, handling of [198](#)
- ERTB list command
 - description of [163](#)
- EXEC initiation outside of ISMF [174](#)
- EXECs
 - initiating [173](#)
 - writing [173](#)

- EXECUTE list command
 - description of [163](#)
- exit
 - (IGDACERO), input variables for the ACS module [208](#)
 - (IGDACERW), values set by the ACS module [213](#)
 - (IGDACSPM), parameter list for the ACS module [207](#)
 - ACS module [203](#)
 - ACS REXX [191](#)
 - activating the REXX [192](#)
 - before expiration [120](#)
 - before migration [120](#)
 - coding the REXX [192](#)
 - errors, ACS module [207](#)
 - installation-wide, using [114](#)
 - REXX, activating [192](#)
 - REXX, coding [192](#)
 - exit routine, coding the module [204](#)
- EXIT statement (ACS)
 - description of [184](#), [189](#)
- exiting ISMF [19](#)
- expiration attributes, management class [39](#)
- expiration, exit before [120](#)
- expressions, boolean [183](#)
- EXTRACT subcommand (DFSMS/VM) [195](#)

F

- failure, recovering from a secondary storage [85](#)
- failure, recovering from an ML1 [85](#)
- failure, recovering from an ML2 [86](#)
- file list panel column variables [235](#)
- file pool, renaming a [77](#)
- file space security [115](#)
- FILEDEF statements [174](#)
- FILELIST line operator
 - description of [166](#)
- FILEPOOL BACKUP command
 - description of [84](#), [85](#)
- files recovery [85](#)
- FILTER list command
 - description of [164](#)
- FILTLIST statement (ACS)
 - description of [184](#), [185](#)
 - EXCLUDE keyword [185](#)
 - INCLUDE keyword [185](#)
- FIND list command
 - description of [164](#)
- FSMMECHK exit [120](#)
- fully qualified LU name [78](#)

G

- general information [203](#)
- grouping minidisks for move requests [93](#)

H

- help
 - facility in ISMF [32](#)
 - for DFSMS commands [133](#)
 - for messages, online [133](#)
 - index [33](#)
 - messages [32](#)

help (*continued*)
panels, task-related [32](#)
HELP list command
description of [164](#)
using [32](#)
HIDE line operator
description of [165](#), [166](#)

I

identification, management class [39](#)
IF statement (ACS)
description of [184](#), [187](#)
IGDACERO, input variables for the ACS module exit [208](#)
IGDACERW, values set by the ACS module exit [213](#)
IGDACSPM, parameter list for the ACS module exit [207](#)
immediate stop [112](#)
index, help [33](#)
indexing function, ACS directory-level [183](#)
initialization [111](#)
initialization and active configurations [10](#)
installation-wide exit usage [114](#)
interface, DFSMS/VM [11](#)
invalid and valid configurations [10](#)
ISearchByKeyCmdACTIVATE [135](#)
ISearchByKeyCmdALTER [136](#)
ISearchByKeyCmdCHECK [138](#)
ISearchByKeyCmdCONVERT [140](#)
ISearchByKeyCmdCOPY [141](#)
ISearchByKeyCmdDELETE [144](#)
ISearchByKeyCmdDFSMS ACTIVATE [135](#)
ISearchByKeyCmdDFSMS ALTER [136](#)
ISearchByKeyCmdDFSMS CHECK [138](#)
ISearchByKeyCmdDFSMS CONVERT [140](#)
ISearchByKeyCmdDFSMS COPY [141](#)
ISearchByKeyCmdDFSMS DELETE [144](#)
ISearchByKeyCmdDFSMS DISCARD [146](#)
ISearchByKeyCmdDFSMS MANAGE [147](#)
ISearchByKeyCmdDFSMS MIGRATE [150](#)
ISearchByKeyCmdDFSMS MOVE [151](#)
ISearchByKeyCmdDFSMS QUERY [157](#)
ISearchByKeyCmdDFSMS RECALL [158](#)
ISearchByKeyCmdDFSMS REPORT [160](#)
ISearchByKeyCmdDFSMS STOP [161](#)
ISearchByKeyCmdDISCARD [146](#)
ISearchByKeyCmdMANAGE [147](#)
ISearchByKeyCmdMIGRATE [150](#)
ISearchByKeyCmdMOVE [151](#)
ISearchByKeyCmdQUERY [157](#)
ISearchByKeyCmdRECALL [158](#)
ISearchByKeyCmdREPORT [160](#)
ISearchByKeyCmdSTOP [161](#)
ISMF
automated move process [98](#)
command status [165](#)
entering [19](#)
exiting [19](#)
for minidisk tasks [87](#)
help facility [32](#)
help messages [32](#)
help panels [32](#)
invoking [19](#)
line operators
ALTER [166](#)

ISMF (*continued*)
line operators (*continued*)
BROWSE [166](#)
CHECK [165](#)
COPY [166](#)
DELETE [166](#)
DISCARD [165](#)
DISPLAY [166](#)
EDIT [166](#)
ERASE [165](#)
FILELIST [166](#)
HIDE [165](#), [166](#)
LIST [166](#)
LISTFILE [166](#)
MESSAGE [165](#)–[167](#)
MIGRATE [166](#)
MOVE [99](#), [165](#)
QUERY [165](#)
RECALL [166](#)
XEDIT [166](#)
list commands
ACTIVATE [163](#)
ALTER FILESPACE [163](#)
BOTTOM [163](#)
CLEAR [163](#)
CP/CMS [163](#)
CURSOR [163](#)
DISCARD [163](#)
DOWN [163](#)
END [163](#)
ERTB [163](#)
EXECUTE [163](#)
FILTER [164](#)
FIND [164](#)
HELP [164](#)
LEFT [164](#)
MIGRATE [164](#)
MOVE [164](#)
PROFILE [164](#)
QUERY [164](#)
RECALL [164](#)
REFRESH [164](#)
RESHOW [164](#)
RETRIEVE [164](#)
RETURN [164](#)
RIGHT [164](#)
SAVE [164](#)
SORT [164](#)
SPLIT [164](#)
STOP [164](#)
SWAP [164](#)
TOP [164](#)
UP [165](#)
VALIDATE [165](#)
user extensions [171](#)
using for ACS tasks [66](#)
variables

column variables for saved lists [239](#)
control variables for File application [233](#)
control variables for ISMF lists [233](#)
control variables for Management Class application [233](#)
control variables for Minidisk application [233](#)
control variables for Saved Lists application [233](#)

ISMF (*continued*)
variables (*continued*)
file list variables [235](#)
management class list variables [237](#)
minidisk list panel column variables [236](#)
Saved ISMF Lists Panel column variables [238](#)

ISMF panels
help [32](#)
moving from panel to panel [21](#)
unique identifiers (panel ID) [21](#)

ISPF log
processing information [170](#)
table services [173](#)

issuing move requests [95](#)

K

key assignments, setting PF [35](#)

KEYS command
description of [36](#)

L

language constants, ACS [177](#)
language statements, ACS [184](#)
large environment, managing [221](#)
last-use mode for multiple entries in a list [168](#)

LEFT list command
description of [164](#)

LIBDEF statements [174](#)

line operator history
ISPF log entries [170](#)
messages [169](#)
obtaining [169](#)
request IDs [110](#)
symbols [169](#)

line operators
descriptions of [165](#), [167](#)
in last-use mode [167](#)
ISMF [165](#)
repeating [167](#)
using [162](#)

List Application [30](#)

list commands
descriptions of [162](#), [165](#)

LIST line operator
description of [166](#)

list printing [32](#)

LISTFILE line operator
description of [166](#)

lists
creation for minidisks [87](#)
ISMF control variables [233](#)
management class [44](#)
minidisk [89](#)

log information, ISPF [170](#)

logon/logoff process for service machines [111](#)

LU name, fully qualified [78](#)

M

MACLIB
defining [174](#)

MANAGE command
description of [147](#)
syntax [147](#)

management classes
application selection panel for storage administrator [43](#)
assigning [42](#), [205](#)
attributes summary [41](#)
define [48](#)
delete [46](#)
display [47](#)
expiration attributes [39](#)
identification [39](#)
list [44](#)
list panel column variables [237](#)
migration attributes [40](#)
preserving names [42](#)
rules and usage of [7](#)
sort entry panel [46](#)
summary of attributes [41](#)

management, minidisk [7](#)

management, space [7](#)

managing
a storage group
sample environment [221](#)
DFSMS/VM storage [117](#)
the large environment [221](#)
the small system [223](#)

mask examples, directory path [178](#)

mask examples, simple [178](#)

mask rules, directory path [178](#)

mask rules, simple [177](#)

message examples, notation used in [xxi](#)

message help [133](#)

MESSAGE line operator
description of [165–167](#)
using [110](#)

messages, help [32](#)

messages, returning [206](#)

MIGRATE command
description of [150](#)
syntax [150](#)

MIGRATE line operator
description of [166](#)

MIGRATE list command
description of [164](#)

migrated data, accessing information [123](#)

migration attributes, management class [40](#)

migration, exit before [120](#)

minidisk list
access [92](#)
creating [87](#)
panel column variables [236](#)
saving [91](#)
working with [87](#)

minidisk management [7](#)

minidisk tasks, using ISMF [87](#)

minidisks
checking [106](#)
creating lists of [87](#)
moving [99](#)
moving to new DASD [93](#)

ML1 failure, recovering from an [85](#)

ML2 failure, recovering from an [86](#)

module exit

- module exit (*continued*)
 - (IGDACERO), input variables for ACS [208](#)
 - (IGDACERW), values set by ACS [213](#)
 - (IGDACSPM), parameter list for ACS [207](#)
 - ACS [203](#)
 - activation of [204](#)
 - coding [204](#)
 - errors, ACS [207](#)
 - routine attributes [203](#)
- move activities [93](#)
- MOVE command
 - description of [151](#)
 - restrictions [155](#)
 - syntax [151](#)
- MOVE line operator
 - description of [165](#)
 - using [99](#)
- MOVE list command
 - description of [164](#)
- move options
 - selecting [94](#)
- move requests
 - issuing [95](#)
 - tracking [95](#)
- moving
 - issuing from ISMF [99](#)
 - selecting options [94](#), [96](#)
- moving CMS minidisks
 - grouping minidisks [93](#)
 - issuing move requests [95](#)
 - selecting move options [94](#)
 - to new DASD [93](#)
 - tracking requests [95](#)
- moving lists [102](#)
- moving through the help panels [33](#)
- multi-language support [255](#)

N

- NDIRS and SIZE [193](#)
- new DASD, moving CMS minidisks to [93](#)
- notation used in message and response examples [xxi](#)

O

- object information, display ACS [72](#)
- operational commands [132](#)
- operators, comparison [182](#)
- operators, line
 - descriptions of [165](#), [167](#)
- operators, using ISMF commands and line [162](#)
- options, selecting move [94](#)
- other ACS variables [194](#)
- output from check request [106](#)
- output listing disposition [73](#)

P

- panel for storage administrator, management class
 - application selection [43](#)
- parameter list for the ACS module exit (IGDACSPM) [207](#)
- parameters, entry [204](#)
- path mask

- path mask (*continued*)
 - examples, directory [178](#)
 - rules, directory [178](#)
- PATH variable (simple) [193](#)
- PATH. variable (stem) [193](#)
- pattern matching [134](#)
- performance [198](#)
- PF keys
 - changing [34](#)
 - defaults [34](#)
 - displaying settings [33](#)
 - setting assignments [35](#)
 - using in ISMF [34](#)
- PFSHOW command [33](#)
- preserving management class names [42](#)
- printing a list [32](#)
- PRNTMINI
 - file tailoring skeleton [241](#)
 - listing [241](#)
 - sample output [247](#)
 - using [171](#)
- PROC statement (ACS)
 - description of [184](#)
- processing, starting and stopping DFSMS/VM [111](#)
- PROFILE list command
 - description of [164](#)
- program function (PF) keys, setting [34](#)
- programming interface information [266](#)

Q

- QUERY command
 - description of [157](#)
 - syntax [157](#)
- QUERY line operator
 - description of [165](#)
- QUERY list command
 - description of [164](#)
- QUIESCE, STOP option [112](#)

R

- RACF/VM for security, using [113](#)
- read-only variable
 - ACS routine [182](#)
- read-only variables, ACS language [179](#)
- read/write variable [194](#)
- read/write variables, ACS language [178](#)
- reason codes from ACS routine [215](#)
- recall by command [122](#)
- RECALL command
 - description of [121](#), [158](#)
 - syntax [158](#)
- RECALL line operator
 - description of [166](#)
- RECALL list command
 - description of [164](#)
- recall, automatic [121](#)
- recovering
 - a file [81](#)
 - an entire storage group [84](#)
 - files using a non-IBM backup product [85](#)

- recovering (*continued*)
 - from a secondary storage failure [85](#)
 - from an ML1 failure [85](#)
 - from an ML2 failure [86](#)
 - specific files [85](#)
- recovery processing [81](#)
- REFRESH list command
 - description of [164](#)
- register contents [206](#)
- renaming
 - a file pool [77](#)
 - the fully qualified LU name [78](#)
 - the global resource ID [78](#)
- repeating a line operator [167](#)
- REPORT command
 - description of [160](#)
 - syntax [160](#)
- reporting on migrated data [123](#)
- request IDs
 - obtaining [110](#)
- requests, grouping minidisks for move [93](#)
- requests, tracking move [95](#)
- RESHOW list command
 - description of [164](#)
- response examples, notation used in [xxi](#)
- RETRIEVE list command
 - description of [164](#)
- return codes [132](#), [206](#)
- return codes from ACS routine [215](#)
- RETURN list command
 - description of [164](#)
 - using [21](#)
- returning control to DFSMS/VM [197](#), [206](#)
- REXX exit
 - ACS [191](#)
 - activating [192](#)
 - coding [192](#)
- REXX/370 compiler [191](#)
- RIGHT list command
 - description of [164](#)
- routine, coding the module exit [204](#)
- routine, invoking the ACS [213](#)
- rules and usage of management classes [7](#)

S

- sample
 - systems [217](#)
 - user extension program [241](#)
- sample environment, ACS routine in a [219](#)
- SAVE list command
 - description of [164](#)
- saved list access through EXECs [173](#)
- saved list sharing [31](#)
- saved minidisk list access [92](#)
- saved minidisk lists
 - accessing through the Minidisk Application [92](#)
 - using EXECs to access [173](#)
- saving a minidisk list [91](#)
- secondary storage failure, recovering from a [85](#)
- secondary storage reporting [123](#)
- secondary storage, specifying no [75](#)
- security
 - considerations [116](#)

- security (*continued*)
 - controls [113](#)
 - of file spaces that DFSMS/VM uses [115](#)
 - using a different facility [114](#)
 - using RACF/VM [113](#)
- SELECT statement (ACS)
 - description of [184](#), [188](#)
- service machines
 - logon/logoff process [111](#)
- SET statement (ACS)
 - description of [184](#), [186](#)
- SET subcommand (DFSMS/VM) [196](#)
- SFS storage groups, converting; sample environment [220](#)
- share data, different active configurations on systems that [10](#)
- shared file system [11](#)
- sharing saved lists [31](#)
- simple mask examples [178](#)
- simple mask rules [177](#)
- SIZE, NDIRS and [193](#)
- slash character (/) [172](#)
- small system, management [223](#)
- sort entry panel, management class [46](#)
- SORT list command
 - description of [164](#)
- source
 - activating a file with ISMF [59](#)
 - invalid configuration [53](#)
 - types of configurations [53](#)
 - valid configuration [53](#)
- source configuration file, deleting an ACS object from [73](#)
- source configuration validation [58](#)
- space management [7](#)
- SPLIT list command
 - description of [164](#)
- starting DFSMS/VM processing [111](#)
- STOP command
 - description of [161](#)
 - syntax [161](#)
- stop immediate [112](#)
- STOP list command
 - description of [164](#)
- stop processing [111](#)
- STOP QUIESCE option [112](#)
- stopping DFSMS/VM processing [111](#)
- stopping DFSMS/VM server virtual machines [111](#)
- stopping the DFSMS/VM master virtual machine [111](#)
- storage administrator, management class application selection panel for [43](#)
- storage groups
 - backed up with the FILEPOOL BACKUP command [84](#)
 - conversion
 - sample environment [220](#)
 - managing
 - sample environment [221](#)
 - recovery [84](#)
- storage, managing DFSMS/VM [117](#)
- subcommand environment, DFSMS [195](#)
- subcommands, DFSMS/VM
 - EXTRACT [195](#)
 - SET [196](#)
 - WRITE [197](#)
- summary of management class attributes [41](#)
- SWAP list command

SWAP list command (*continued*)
description of [164](#)
symbols
question mark (?) [169](#)
syntax diagrams, how to read [xix](#)
systems, sample [217](#)

T

target minidisk size [98](#), [101](#), [104](#)
termination [111](#)
test ACS routines [71](#)
test case define or ACS test case alter, ACS [69](#)
test listing, ACS [71](#)
test selection panel, ACS [68](#)
TOP list command
description of [164](#)
tracking move requests [95](#)
translate ACS routines [67](#)
translator listing, ACS [68](#)

U

UP list command
description of [165](#)
user authorization and security controls [113](#)
user extensions
ISMF variables [233](#)
sample of [241](#)
using [171](#)
user extensions, reference [171](#)

V

valid and invalid configurations [10](#)
VALIDATE list command
description of [165](#)
validating the source configuration [58](#)
validation listing [59](#)
values set by the ACS module exit (IGDACERW) [213](#)
variables
ACS [192](#)
ACS language read-only [179](#)
ACS language read/write [178](#)
column variables for saved lists [239](#)
control variables for file application [233](#)
control variables for ISMF lists [233](#)
control variables for management class application [233](#)
control variables for minidisk application [233](#)
control variables for saved lists application [233](#)
file list panel column [235](#)
file list variables [235](#)
management class list variables [237](#)
minidisk list panel [236](#)
other ACS [194](#)
PATH (simple) [193](#)
PATH. (stem) [193](#)
read-only [192](#)
read/write [194](#)
saved ISMF lists panel column variables [238](#)

W

wild cards [134](#)
WRITE statement (ACS)
description of [184](#), [189](#)
WRITE subcommand (DFSMS/VM) [197](#)
writing EXECs [171](#)

X

XEDIT line operator
description of [166](#)



Product Number: 5741-A09

Printed in USA

SC24-6279-74

