CA Mainframe Network Management

Reference Guide

r12

This documentation and any related computer software help programs (hereinafter referred to as the "Documentation") are for your informational purposes only and are subject to change or withdrawal by CA at any time.

This Documentation may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Documentation is confidential and proprietary information of CA and may not be used or disclosed by you except as may be permitted in a separate confidentiality agreement between you and CA.

Notwithstanding the foregoing, if you are a licensed user of the software product(s) addressed in the Documentation, you may print a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all CA copyright notices and legends are affixed to each reproduced copy.

The right to print copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO THE END USER OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and is not modified in any way by the terms of this notice.

The manufacturer of this Documentation is CA.

Provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

Copyright © 2010 CA. All rights reserved. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

CA Product References

This document references the following CA products:

- CA 7 Workload Automation (CA 7)
- CA Auditor
- CA Automation Point
- CA Datacom® (CA Datacom)
- CA Mainframe Software Manager (CA MSM)
- CA NetMaster® File Transfer Management (CA NetMaster FTM)
- CA NetMaster® Network Automation (CA NetMaster NA)
- CA NetMaster® Network Management for SNA (CA NetMaster NM for SNA)
- CA NetMaster® Network Management for TCP/IP (CA NetMaster NM for TCP/IP)
- CA NetMaster® Socket Management for CICS (CA NetMaster SM for CICS)
- CA NetSpy™ Network Performance (CA NetSpy)
- CA SOLVE:Access® Session Management (CA SOLVE:Access)
- CA SOLVE:Central™ Service Desk for z/OS (CA SOLVE:Central)
- CA SOLVE:FTS
- CA SOLVE:Operations® Automation (CA SOLVE:Operations Automation)
- CA SOLVE:Operations® Automation for CICS (CA SOLVE:Operations Automation for CICS)
- CA TCPaccess™ Communications Server (CA TCPaccess CS)

Note: This guide contains descriptive text and procedures about options and products that you may not be licensed for or have not enabled. Inclusion of the descriptions of these options and products in this guide in no way implies that you are licensed for these options or products.

Contact CA

Contact Technical Support

For your convenience, CA provides one site where you can access the information you need for your Home Office, Small Business, and Enterprise CA products. At http://ca.com/support, you can access the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Provide Feedback

If you have comments or questions about CA product documentation, you can send a message to techpubs@ca.com.

If you would like to provide feedback about CA product documentation, complete our short customer survey, which is also available on the CA Support website, found at http://ca.com/docs.

Contents

Chapter 1: Introduction 19

Intended Audience 19
Typographic Conventions 19

Chapter 2: Advanced Customization Tasks 21

Customize OCS Function Key Settings 21
List Function Key Settings 21
Set Local Function Keys 21
Set Global Function Keys 22
Default OCS Function Key Settings 22
Equate Command Strings 23
Replace Commands After Exit Initialization 23
Implement the Time-out Facility 23
Customize National Language Character Set Support 24
What National Language Support Affects 24
Determine a User's Language Code 25
Provide a Customized Primary Menu 25
Supplied Primary Menu Procedure 25
Display the Primary Menu 26

Chapter 3: Initializing Your Region Using the NETMASTR Program 29

About NETMASTR 29
How NETMASTR Processing Works 29
Specify the NMDRVCTL Data Set 30
NETMASTR Commands 31
General Commands 31
Dynamically Allocate a Data Set 32
Dynamically Allocate a SYSOUT Data Set 34

Chapter 4: Tuning Performance 35

Enhance Performance 35
Tune at the System Level 36
Tune VTAM Interface 36
Tune Panel Use 37
Tune Storage Limits for Panel Sends 37
Improve NCL Procedure Usage 38

Monitor NCL Procedure Loading Activity . . . 39
Preload NCL Procedures . . . 39
Tune VSAM Data Sets . . . 39
Buffer Sharing . . . 40
Deferred Write Capabilities . . . 40
VSAM Processing in a Subtask . . . 41
Record Size Management . . . 42
Database Activity Management . . . 43
Customize VSAM Alerts . . . 43
Performance Monitoring Facilities . . . 44
Tune Communication Between Regions . . . 46
Tune INMC . . . 46
Tune ROF . . . 47
Tune CA SOLVE:FTS . . . 48
Tune ISR . . . 48
Control Message Flow in OCS . . . 49
Performance Considerations When Writing NCL . . . 49
Use Commands to Control Resource Consumption by NCL . . . 50
Performance and Tuning Commands . . . 50
SHOW SYSPGT . . . 51
SHOW SYSRCT . . . 53
SHOW SYSWAIT . . . 56
SYSPGT . . . 58
SYSRCT . . . 58
Record CPU Usage . . . 61
NCL CPU-time Accounting . . . 61
Implement NCL CPU-time Accounting . . . 61
Start and Stop CPU-time Accounting . . . 63
Display the Status of NCL CPU-time Accounting . . . 65

Chapter 5: Using the SYSCMD Facility 67

SYSCMD Facility . . . 67
Set Console Time-outs . . . 67
Lock Consoles to a Specific Environment . . . 68
Simulate Command Issue from a Specific Console . . . 69
Display the Status of the Console Pool . . . 70
Use SYSCMD in an &INTCMD Environment . . . 70
Clear a Lock in an &INTCMD Environment . . . 71
Prefix Messages . . . 71
Facilitate User-Generated Messages . . . 71
Insertion Points . . . 72
&WTO NCL Statement . . . 72

&WTOR NCL Statement . . . 73
&DOM NCL Statement . . . 74
SYSCMD Facility Commands . . . 75
SYSCMD NCL Verb and System Variable Summary . . . 75

Chapter 6: Administering Signed-on Users 79

Show Active Users . . . 79
SHOW USERS Command . . . 79
Active User List Facility . . . 79

Chapter 7: Customizing External Applications Access 81

External Applications Function . . . 81
Access External Applications Function . . . 82
Access Applications . . . 83
Navigate to an External Application . . . 83

Chapter 8: Customizing and Using MAI-OC 85

MAI-OC . . . 86
Define MAI-OC to Target Applications . . . 87
Sample LU Definitions . . . 87
Cross-Domain MAI-OC Sessions . . . 90
Log On to Another Application . . . 90
Create a Session Identifier . . . 91
List Established Sessions . . . 91
Select an LU Name for an MAI-OC Session . . . 91
Log Off an Application . . . 93
Disconnect an MAI-OC Session . . . 93
Interrupt an MAI-OC Session . . . 93
Send Data to an Application . . . 94
Receive Data from an Application . . . 94
Issue Commands to an Application . . . 95
Issue Commands While Waiting for Application Response . . . 95
Abbreviate Commands . . . 95
Issue Multi-Segment Commands to an IMS Application . . . 96
Use MAI-OC with Multiple Regions . . . 96
Set EQUATE Values for MAI-OC Commands . . . 97
Use an MAI Installation Exit . . . 98
Session Protocols . . . 98
SCS Character Support . . . 99
SCS Characters Sent by MAI-OC . . . 101
Strike-over Masks . . . 101

JES MAI-OC Sessions . . . 102
MAI-OC Mode Table and Bind Checks . . . 102
MAI-OC Logmode Entry Selection . . . 103
MAI-OC Operational Scenario . . . 104
Production Machine Definitions . . . 104
Network Operator Action . . . 105
Other User . . . 106
Testing Machine Definitions . . . 106
User Action . . . 107

Chapter 9: Working With System Image Definitions 109

Implement System Images . . . 110
Access System Image Definitions . . . 111
Define a System Image . . . 112
Maintain System Images . . . 113
Merge System Images . . . 113
Add Resource Definitions to a System Image . . . 113
Why Do You Need to Define Resources? . . . 114
Access Resource Definitions . . . 114
Access Resource Definitions from the Resource Definition Panel . . . 114
Access Resource Definitions from the System Image List . . . 114
Access Resource Definitions from the Status Monitor . . . 115
Define a Resource to a System Image . . . 115
Describe the Resource . . . 115
Define the Activation Details . . . 118
Define the Inactivation Details . . . 124
Define the Forced Inactivation Details . . . 126
Define the Display and Heartbeat Details . . . 128
Define the Status Monitor Message Details . . . 130
Implement the State Change Exits . . . 133
Define the Logging Details . . . 133
Specify the Owner Details . . . 135
Implement the Extended Function Exit . . . 136
Time-outs . . . 137
Specify Messages in a Resource Definition . . . 138
Select Messages in the Transient Log . . . 139
Special Message Prefixes . . . 140
Extend the Definition of Resource Message Rules . . . 141
Define an Extended Message Filter . . . 141
Define Event-related Actions . . . 142
Implement Event Exits . . . 142
Define Extended Display Attributes . . . 143

Document the Message Rule . . . 145
Execute an NCL Procedure . . . 146
Create User-defined Resource Subclasses . . . 147
Logical Resources . . . 148
Define Resource Relationships . . . 148
Primary and Alternate Relationships . . . 150
Effect of Resource Relationships on Operations . . . 150
Effect of a Resource Set to the OFF Operation Mode on a Relationship . . . 151
Staged Image Load and Shutdown . . . 152
System Load Balancing . . . 152
How MV and MVT Commands Move the Operation of a Resource . . . 153
Define Shared Resources in Local System Images . . . 154
Resource Definition Maintenance . . . 155

Chapter 10: Backing Up the RAMDB 157

Database Maintenance . . . 157
Unlinked Non-production Regions . . . 157
Unlinked Production Regions . . . 158
Linked Regions . . . 159

Chapter 11: Reporting on Your RAMDB 161

About Reporting . . . 161
Generate a Report . . . 162
Search for Macros . . . 163
Search for Message Rules . . . 163
Search for Processes . . . 164
Use Your Search Output . . . 164

Chapter 12: Defining Macros 167

Understanding Macros . . . 167
NCL Procedures as Macros . . . 167
Customize the $RMMC00S Macro Template . . . 168
Customize the $RMMC00D Macro Template . . . 168
Register and Maintain the Macros . . . 168
Access Macro Definitions . . . 169
Macro Definition Panel . . . 169
Add and Maintain Macro Definitions . . . 169

Chapter 13: Defining Automation Services Commands 171

Automation Services Commands . . . 171

Register and Maintain Commands . . . 171
Access Command Definitions . . . 172
Command Details . . . 173
Command Prompting, Confirmation, and Validation . . . 174
Add and Maintain Command Definitions . . . 175
Define Your Own Commands . . . 175
Code Your NCL Procedure . . . 176
Variables Available to a Command NCL Procedure . . . 176

Chapter 14: Customizing the Display Attribute Tables 179

Display Attribute Tables . . . 179
Edit a Display Attribute Table . . . 180
Logical State Attributes Table . . . 181
Logical State Attributes and SNA Network Summary Display . . . 182
Automated Mode Attributes Table . . . 182
Set the Logical States for the Automated Mode . . . 183
Manual Mode Attributes Table . . . 183
Set the Logical States for the Manual Mode . . . 184
SNA Resource Logical State Normalization Table . . . 185
Set the Logical States . . . 186

Chapter 15: Customizing the Status Monitor Display Format 187

Status Monitor Display Formats . . . 187
Status Monitor Views . . . 188
Multi-column Displays . . . 188
Create a Status Monitor Display Format . . . 189
Specify the Status Monitor Display Format . . . 191
Specify a Multiscreen Display Format . . . 194

Chapter 16: Maintaining Prompt Lists 195

Prompt Lists . . . 195
Access Prompt Lists . . . 195
Add a Value to a Prompt List . . . 196
Update Prompt List Definitions . . . 198

Chapter 17: Using Linked Product Regions 199

Communicate Between Your Product Regions . . . 199
Display Links . . . 200
Link Regions . . . 200
Inter-Network Management Connection . . . 200

Access Methods . . . 202
Security . . . 202
Traffic Flow . . . 202
Control Links . . . 203
INMC Links . . . 203
Static INMC Links . . . 203
Dynamic INMC Links . . . 204
Establish INMC Links . . . 204
Routing Commands . . . 205
Communication Recovery . . . 205
Control INMC Links . . . 206
Start an INMC Link . . . 206
Stop an INMC Link . . . 206
Specify a Particular Route for INMC Traffic . . . 207
Reset an INMC Link . . . 207
Display INMC Links . . . 208
Improve INMC Link Performance . . . 208
Use Security with INMC Links . . . 209
Diagnose Problems with INMC Connections . . . 209
Troubleshooting for INMC Links . . . 210
Plan INMC Links . . . 211
Plan Simple-mode Links . . . 211
Plan Rotate and Backup Mode Multipath Links . . . 213
Plan Preferential Mode Multipath Links . . . 215
Maintain Link Definitions . . . 216
Display INMC Link Definitions . . . 216
Create an INMC Link Definition . . . 216
Modify an INMC Link Definition . . . 218
Delete an INMC Link Definition . . . 219
Provide Centralized Control of Connected Regions . . . 219
Use ROF for Centralized Control of Remote Regions . . . 219
Sign on to a Remote Region . . . 221
Issue Commands on a Remote Region . . . 222
Receive ROF Messages . . . 223
Sign Off a Remote Region . . . 224
Use ISR for Centralized Control of Remote Domains . . . 224
Manage APPC Links . . . 229
Start APPC Links . . . 229
Display APPC Link Status . . . 230
Stop APPC Links . . . 231

Chapter 18: Broadcasts 233

Broadcast Services . . . 233
Types of Broadcasts . . . 234
List Broadcasts . . . 234
View Active Broadcasts . . . 234
Delete Active Broadcasts . . . 234
Broadcast to Generic Resources . . . 235
Enable Broadcasts to Generic Resources . . . 235
Add Resources to a Broadcast System Group . . . 236
Send Broadcasts to Generic Resources . . . 237
Set General Broadcasts . . . 238
Set Primary Menu Broadcasts . . . 239
Review Active Broadcasts . . . 239
Send Dynamic Broadcasts . . . 240
Enter Broadcast Text . . . 240
Receive Broadcasts . . . 241
Access System Group Definition . . . 241

Appendix A: Connecting Terminals 243

Supported Terminals . . . 243
How Your Product Accepts Terminal Connections . . . 244
Extended Attributes Support . . . 244
Specify Screen Sizes . . . 245
Specify Logmode Table Definitions . . . 246

Appendix B: Product Region JCL Parameters 247

Product Region JCL Parameter Descriptions . . . 247
Product Name Keys . . . 254

Appendix C: SYSPARMS Operands 257

Generic SYSPARMS Summary Table . . . 257
SYSPARMS Operand Descriptions . . . 264

Appendix D: MAI Installation Exit MAIEX02 295

MAI Installation Exit . . . 295
Implement the MAI Exit . . . 296
Reentrant Code . . . 296
Storage Subpools . . . 296
Exit Serialization . . . 297
Sample Exit . . . 298

Start the Exit . . . 298
Registers on Entry to the Exit . . . 299
Exit Correlators . . . 299
System Correlator . . . 299
User Correlator . . . 299
Session Correlator . . . 299
Security Exit Correlator . . . 300
Security Exit User Token . . . 300
Contents of the Communications Area . . . 300
Exit Initialization Call . . . 300
Exit Termination Call . . . 301
MAI Session Start Call . . . 302
ACB Open Call . . . 303
MAI Session End Call . . . 304
Return Codes from the Exit . . . 304
Exit Initialization . . . 304
Session Start . . . 305

Appendix E: Consoles and Migration ID Exits 307

Console Use . . . 307
JES, OP1, OP2, and Pseudo Consoles . . . 307
Extended MCS Consoles . . . 308
Migration IDs . . . 309
Console Management . . . 309
Console Pool Management . . . 310
Specify the Number of Consoles . . . 310
Considerations . . . 311
Use Migration ID Exits to Customize Console Management . . . 311
Allow Dynamic Determination of the Need for Migration IDs . . . 311
User Exits for Migration IDs . . . 312
Why Define a User Exit? . . . 312
Sample Exit . . . 312
Activate the User Exit . . . 312
User Exit Requirements . . . 313
Input Parameters for User Exits . . . 313
Return Codes from a User Migration ID Exit . . . 315
Define Consoles to a CICS Region . . . 315

Appendix F: Automation Services Application Program Interface 317

Application Programming Interface Procedures . . . 318
$RMCALL API . . . 319

$RMDBAPI API . . . 320
$RMEVENT API . . . 321
$RMSTSET API . . . 321
$RECALL API . . . 322
$REDBAPI API . . . 323
$RMCALL ACTION=COMMAND . . . 324
Supplied Commands That Require Parameters . . . 327
$RMCALL ACTION=DBGET . . . 329
$RMCALL ACTION=PURGE . . . 331
$RMCALL ACTION=STGET . . . 332
$RMDBAPI SERVICE=CREATE . . . 334
Specify Field Values When Calling $RMDBAPI from an NCL Procedure . . . 337
Specify Field Values When Submitting $RMDBAPI as a Command . . . 338
ResourceView Definition Field Names . . . 339
$RMDBAPI SERVICE=DELETE . . . 347
$RMDBAPI SERVICE=GET . . . 349
$RMDBAPI SERVICE=LIST . . . 352
$RMDBAPI SERVICE=SET . . . 354
$RMEVENT . . . 356
$RMSTSET . . . 357
$RECALL SERVICE=SET . . . 358
$RECALL SERVICE=GET . . . 360
$RECALL SERVICE=ACTION . . . 362
$REDBAPI SERVICE=CREATE . . . 363
Specify Field Values When Calling $REDBAPI from an NCL Procedure . . . 365
Specify Field Values When Submitting $REDBAPI as a Command . . . 366
EventView Definition Field Names . . . 367
$REDBAPI SERVICE=DELETE . . . 370
$REDBAPI SERVICE=GET . . . 372
$REDBAPI SERVICE=LIST . . . 374

Appendix G: Automation Services SMF Record Format 377

Automation Services SMF Records . . . 377
SMF Header Format . . . 378
How to Obtain User SMF Record Types . . . 378
EventView SMF Record Format . . . 379
ResourceView and ServiceView Record Format . . . 380
User-defined Record Format . . . 381

Appendix H: RAMDB Definition Classes 383

Definition Classes . . . 383

Appendix I: RAMDB Variables 385

Variable Types . . . 385
Knowledge Base Variables . . . 386
Description Panel Fields . . . 387
Activation and Inactivation Details Fields . . . 388
Display and Heartbeat Field . . . 388
Automation Log Details Fields . . . 389
First Level Support Details Fields . . . 389
Second Level Support Details Fields . . . 390
Status Variables . . . 390
Message Variables . . . 393

Appendix J: Process Macros 397

Macros . . . 398
AOMALERT Macro . . . 398
CHAIN Macro . . . 398
COMMAND Macro . . . 400
COMPARE Macro . . . 403
CONCAT Macro . . . 405
DELALERT Macro . . . 406
EVENT Macro . . . 406
EVVARGET Macro . . . 409
EVVARSET Macro . . . 410
EXECNCL Macro . . . 411
EXTRACT Macro . . . 412
GENALERT Macro . . . 415
GETSTAT Macro . . . 417
GLBLSAVE Macro . . . 420
GOTO Macro . . . 421
PARSE Macro . . . 421
PINGCD Macro . . . 425
REPLY Macro . . . 426
RUNPRCSS Macro . . . 429
SETRC Macro . . . 430
SETSTATE Macro . . . 430
SETVARS Macro . . . 434
SHDCALL Macro . . . 434
SMFWRITE Macro . . . 436
SNMPTRAP Macro . . . 437
SOCKCLSE Macro . . . 438
SOCKCONN Macro . . . 439

SOCKRECV Macro . . . 440
SOCKSEND Macro . . . 442
STARTNCL Macro . . . 442
STOP Macro . . . 444
SUBJOB Macro . . . 445
SUBPRCSS Macro . . . 448
SVAPI Macro . . . 449
SVCMD Macro . . . 450
SYSCMD Macro . . . 451
TRANS Macro . . . 455
WAIT Macro . . . 457
WAITEVNT Macro . . . 457
WAITSTAT Macro . . . 460
WTO Macro . . . 464
WTOR Macro . . . 465

Appendix K: Data Set Descriptions 467

Data Set Types . . . 467
Installation . . . 469
Product Components . . . 470
Management Services Data Sets . . . 470
PDSE Services Data Sets . . . 476
Automation Services Data Sets . . . 477
SOLVE Subsystem Interface Data Sets . . . 477
TCP/IP Services Data Sets . . . 478
File Transfer Services Data Sets . . . 479
SNA Services Data Sets . . . 482
SNA Automation Services Data Sets . . . 484
FTS Data Sets . . . 485
ReportCenter Data Sets . . . 487
WebCenter SDK Data Sets . . . 487

Appendix L: Distributed Knowledge Base Changes 489

$TEMPLAT 0001 Definitions . . . 489
r12 $TEMPLAT 0001 Definition Changes . . . 490
r11.6 SP1 $TEMPLAT 0001 Definition Changes . . . 490
r11.6 $TEMPLAT 0001 Definition Changes . . . 490
$TEMPLAT 0002 Definitions . . . 492
r12 $TEMPLAT 0002 Definition Changes . . . 493
r11.6 SP1 $TEMPLAT 0002 Definition Changes . . . 493
r11.6 $TEMPLAT 0002 Definition Changes . . . 493

$TEMPLAT 0003 Definitions . . . 495
r11.6 $TEMPLAT 0003 Definitions . . . 496
Commands . . . 496
r12 Command Changes . . . 497
r11.6 SP1 Command Changes . . . 498
r11.6 Command Changes . . . 499
File Transfer Rules . . . 499
r11.6 File Transfer Rule Changes . . . 500
Global Processes . . . 500
r11.6 Global Process Changes . . . 500
IP Node Monitor Groups . . . 500
r11.6 IP Node Monitor Group Changes . . . 501
Macros . . . 501
r11.6 Macro Changes . . . 501
Monitoring Attributes . . . 501
r12 Monitoring Attribute Changes . . . 502
r11.7 Monitoring Attribute Changes . . . 503
r11.6 SP1 Monitoring Attribute Changes . . . 503
r11.6 Monitoring Attribute Changes . . . 505
Prompt Lists . . . 508
r12 Prompt List Changes . . . 508
r11.6 Prompt List Changes . . . 508
Alert Monitor Filters . . . 509
r12 Alert Monitor Filter Changes . . . 509
Status Monitor Filters . . . 509
r12 Status Monitor Filter Changes . . . 509
r11.6 SP1 Status Monitor Filter Changes . . . 510
r11.6 Status Monitor Filter Changes . . . 510
VTAM States . . . 510
r11.6 VTAM State Changes . . . 510
Resource Groups . . . 511
r12 Resource Group Changes . . . 511
Icon Panels . . . 511
r12 Icon Panel Changes . . . 511
r11.6 SP1 Icon Panel Changes . . . 512
r11.6 Icon Panel Changes . . . 512

Index **513**

Chapter 1: Introduction

This section contains the following topics:

Intended Audience (see page 19)
Typographic Conventions (see page 19)

Intended Audience

This guide is intended for technical personnel responsible for the planning and maintenance of your product's functions and services. It contains information about the advanced functionality of your product.

Typographic Conventions

This section explains the conventions used when referring to various types of commands and when indicating field attributes.

Convention	Description
Commands	Commands such as SYSPARM and SHUTDOWN are shown in upper case.
User Entries	Information to enter onto panels is displayed in **bold** text.
Cross-References	Cross-reference links to other sections of the book are displayed as underlined blue text.
Shortcuts	Shortcuts to menus or options are displayed in **bold**, for example, **/PARMS**.

Chapter 2: Advanced Customization Tasks

This section contains the following topics:

Customize OCS Function Key Settings (see page 21)
Equate Command Strings (see page 23)
Implement the Time-out Facility (see page 23)
Customize National Language Character Set Support (see page 24)
Provide a Customized Primary Menu (see page 25)

Customize OCS Function Key Settings

There is support for up to 24 function keys in Operator Console Services (OCS) mode. You can customize these keys either locally or globally.

List Function Key Settings

Use the PF LIST command to list the current settings of the function keys.

Set Local Function Keys

You can customize the function keys of each OCS panel with the PF command. Function keys set from an OCS panel:

- Apply to the requesting panel only
- Are valid only until you exit from OCS

Example

To define F4 as a conversational function key, issue the following command:

```
PF4 CONV,MSG USER1
```

Set Global Function Keys

You can use the PF command to set the default or global values of the function keys that apply on initial entry to OCS. Function keys set in this way apply to all OCS panels unless overridden by a further PF command entered from OCS mode.

To do this, execute the PF command in the BSYS environment (for example, in your INIT or READY procedures) or use the SUBMIT command to pass a PF command to the BSYS environment.

Note: For information about the PF command, see the online help.

Default OCS Function Key Settings

If no PF commands are included in your INIT procedure, the following defaults apply:

Key	Command Issued	Type
F1/13	-FSPROC ##HELP	Prefix
F2/14	SPLIT	Immediate
F3/15	X	Immediate
F4/16	RETURN	Immediate
F5/17	-FSPROC $CMDENT+	Prefix
F6/18	AUTOHOLD	Immediate
F7/19	-FSPROC $LOBROW	Immediate
F8/20	CLEAR	Immediate
F9/21	SWAP	Immediate
F10/22	CS+	Immediate
F11/23	CS-	Immediate
F12/24	ORDER	Immediate

Equate Command Strings

The EQUATES parameter group lets you substitute a long command string with a single one- to eight-character string. The string is then easier and faster to use. This is particularly useful when you are using Remote Operator Facility (ROF). You can also use the EQUATES parameter group to replace an existing command with a Network Control Language (NCL) procedure.

You can set up equates in the EQUATES parameter group when you first customize your region. These equates are then immediately available to any operator using OCS mode. These equates are known as global EQUATES.

Individual users can override a global equate by using the EQUATE command in OCS. Equate values set by an individual user are effective only for that user, and only for the single OCS session.

Note: For more information about the EQUATE command, see the online help.

Replace Commands After Exit Initialization

To substitute commands with NCL procedures after initialization, you can use either the CMDREPLS or EQUATES parameter group.

Note: Command replacement using the EQUATES parameter group takes effect during system initialization only.

Implement the Time-out Facility

You can set terminals that are logged onto an application to time out after a period of inactivity. This reduces the security risk of having them logged on but unattended.

A general time-out facility is provided in your product region. Use this to specify time-out intervals and actions for all terminals.

The time-out facility identifies a user at a terminal as having been inactive for a period and executes an action. Possible actions are to ring the terminal alarm, or to log the user off the system.

When time-out management is implemented, it affects all of the users of your region, unless their user ID definitions exempt them.

The TIMEOUT parameter group lets you administer the time-out facility. You can access parameter groups through the /PARMS panel shortcut.

Customize National Language Character Set Support

Your product supports the use of National Language (NL) character sets. This support is automatic and is controlled by the SYSPARMS LANG command. The default is US; however, many different languages are supported.

Note: For more information about supported language codes and their associated code pages, see the *Network Control Language Reference Guide.*

You can set an individual language code for users in their user ID definitions.

Note: For information about how to specify a language code in a user ID, see the *Security Guide*.

Language codes can also be set in NDBs. These NDBs use the associated code page to perform uppercase folding for storing and searching fields. If the specified language code is not supported, the system code, as set by the SYSPARMS LANG command, is used.

What National Language Support Affects

The following functions perform language support translation using the code page associated with a user's language code:

- NCL panel processing using #FLD CAPS=YES
- NCL statements using the &TRANS built-in function with the NLUPPER and NLLOWER operands

The following functions perform language support translation using the code page associated with the NDB:

&NDBSCAN

Scan expressions, which require data to be folded for CAPS=SEARCH fields, use the code page of the NDB.

&NDBADD

Fields use the code page of the NDB.

&NDBPUT

When these statements add data to fields defined as CAPS=YES, the translation is performed using the code page associated with the NDB.

Determine a User's Language Code

The &ZUSERSLC system variable returns the system recognized language code for a user. This is one of the following values:

- The user's language code
- The system language code if the user's language code is not valid
- The value UK if the system language code is not a supported code

Provide a Customized Primary Menu

If you want to customize the primary menu for your installation, you must understand how the supplied primary menu procedure works.

Note: If you want to customize the supplied procedure, ensure that you rigorously test any changes for all user classes before implementing the new procedure.

When you create a new primary menu procedure, specify the name of the new procedure in the PMENUCONTROL parameter group. You can also specify the name using the SYSPARMS MENUPROC command and include it in your INIT procedure.

Supplied Primary Menu Procedure

The primary menu procedure provides an entry point for every user and provides an appropriate selection list of functions for the users of your system.

The name of the supplied primary menu procedure is $NMPMENU. This procedure is invoked by the system to perform system checks and then invokes the procedure, $NMPEXIT, to perform the presentation of the menu.

Both $NMPMENU and $NMPEXIT are standard NCL procedures and are invoked according to each user's NCL library specification.

The primary menu procedure is invoked under the following conditions:

- After initial logon
- At the termination of an OCS or MAI session
- If the procedure terminated without specifying a mode change or window termination

Expired Password Procedure

$NMPMENU must be able to detect when a user's password has expired. The &ZPWSTAT system variable is used for this purpose. The variable invokes the $UAPWD01 procedure to prompt for password change. Make sure that this facility is available if you customize the supplied procedure.

Reconnection Menu

$NMPMENU supplies a reconnection menu, providing multiple-signon users with the following choices of reconnection options after a session outage:

- Reconnect to a particular region
- Bypass reconnection and display the primary menu
- Bypass reconnection and cancel all disconnected regions

Display the Primary Menu

$NMPEXIT is invoked by $NMPMENU to display the primary menu. The menu panel has the following features:

- Fixed title line
- Input option line
- Error message line
- Floating trailer containing the menu broadcast and available function keys
- 55 lines available for menu selections
- A user information box on the right hand side

A maximum of 12 menu options can be displayed on the menu at one time. The format and number of these selections should be displayable on all devices supported by your installation's network.

$NMPEXIT uses the security query capability of &SECCALL to determine users' authorized privileges. It then matches this profile to determine which features they are authorized to access, and displays only these features on the menu.

Alter the Primary Environment

When the primary menu is displayed, the primary environment is operating in base NCL mode. There are two other modes of primary environment:

- OCS
- MAI-FS

These modes are invoked when the primary menu procedure issues a SETMODE command to modify the primary environment. When these alternate modes are terminated, the primary environment returns to base NCL mode.

For example, if a user is authorized to access OCS and enters the O option on the primary menu, OCS is invoked using the SETMODE OCS command. This alters the primary environment so that it operates in OCS mode. When users exit OCS, they are returned to the primary menu, and the primary environment operates in base NCL mode.

Support Single Option Users

If a user is only authorized for one option on the primary menu, the primary menu procedure can detect this situation and will automatically select that option and display the menu for that option. For example, if a user is only authorized for OCS, the primary menu procedure will bypass the primary menu and change to OCS mode automatically. When the user exits the option, the session is closed without displaying the primary menu.

Chapter 3: Initializing Your Region Using the NETMASTR Program

This section contains the following topics:

About NETMASTR (see page 29)
How NETMASTR Processing Works (see page 29)
Specify the NMDRVCTL Data Set (see page 30)
NETMASTR Commands (see page 31)

About NETMASTR

NETMASTR is a program that lets you do the following:

- Circumvent the 100-character JCL parameter limit.
- Start a region before JES without cataloging data sets in the master catalog.
- Use the MVS system symbol substitution service in a control file to tailor program parameters. This aids in cloning systems in a sysplex environment.

The NETMASTR program is supplied in object code and has the following product-dependent aliases:

- CONNECTR
- SOLVE

How NETMASTR Processing Works

The NETMASTR program processes in the following sequence:

1. The NMDRVCTL data set (the control file) opens.
2. Input commands are read and processed.
3. The NMDRVCTL data set closes.
4. If no errors are detected, control is transferred to the target program (using XCTL), passing the input JCL PARM as prefixed, suffixed, or both.

If errors are detected in the input commands or if a dynamic allocation error occurs, the error message is written to the console as a WTO message and the action specified by the ERROR command is taken. The target program is not started in this case.

Specify the NMDRVCTL Data Set

When NETMASTR executes, it opens and reads the data set associated with the NMDRVCTL ddname to obtain parameters. This section describes how to specify the contents of NMDRVCTL.

The NMDRVCTL DD statement points to the data set that contains control statements for NETMASTR. These control statements specify:

- NETMASTR options
- PARM information for the target program
- Data set or SYSOUT allocation requests

The data set must have the following attributes:

- Fixed (blocked or unblocked)
- LRECL 80
- Any valid block size

The data set can be sequential or a partitioned data set (PDS) member, or a concatenation of these. For example, it can be a member in SYS1.PARMLIB.

The control statements must be in the following format:

- All control statements must be in uppercase.
- Only columns 1 through 72 are examined. If line numbers are desired, they can be in columns 73 through 80.
- Blank lines are ignored.
- Any line with an asterisk (*) as the first non-blank character is treated as a comment and ignored.
- Other lines must contain valid NETMASTR commands (see the following section for details of these commands).

  Only one command is allowed per line. Commas can separate the operands of a command. Comments can trail on the lines if they are separated from the commands and operands by a blank.
- The commands for data set and SYSOUT allocation can span several lines. In this case, the plus sign can be used as a parameter to indicate that the next line is a continuation of the current command.

NETMASTR Commands

The NETMASTR program has three groups of commands:

- General
- The dynamic allocation of data sets
- The dynamic allocation of SYSOUT data sets

General Commands

The following general commands can be specified in the NMDRVCTL data set:

PGM=*name*

Specifies the target program to which control is transferred. If not specified, defaults to NM001.

ERROR={ *tnnnn* | R100 }

Specifies the action to take if an error is detected in the input control statements. The following actions are valid:

R*nnnn*

Returns to the system with *nnnn* return code.

U*nnnn*

Abends with user ABEND code *nnnn*.

Default: R100

PARMSEP=*c*

Specifies the separator character for concatenation of PARM information. The nominated character is used only between the PARM prefix (see the PPREF command), the JCL PARM, and the PARM suffix (see the PSUFF command). The character is not used between individual sections of the prefix and/or suffix.

Default: comma (,)

PPREF=*value*

Specifies the prefix of the supplied JCL parameters that is passed to the target program. This command must not span lines, but can be specified several times. The multiple specifications are concatenated together, in appearance order, with no intervening separators, and treated as a single prefix.

Limits: 1024 characters

PSUFF=*value*

Specifies the suffix of the supplied JCL parameters that is passed to the target program. This command must not span lines, but can be specified several times. The multiple specifications are concatenated together, in appearance order, with no intervening separators, and treated as a single suffix.

Limits: 1024 characters

LIST={ NO | YES }

Controls the listing of input lines. If LIST=YES is specified, all following lines are listed to the console using WTO messages. If SUBS=YES is in effect, each line is also displayed after substitution processing.

SUBS={ NO | YES }

Controls symbolic substitution for line processing. Specifying SUBS=YES enables substitution on all following input lines (except comments).

Specifying SUBS=NO stops substitution on all following input lines.

VAR*xxx*=*'value'*

Lets you define up to 20 additional user variables for substitution. *xxx* is the variable name, and the value is specified after the equal sign (=), which can be quoted if containing blanks. Each variable name can have up to 16 characters.

If substitution is in effect, the variable name and value can be built from other variables.

Note: The value length cannot exceed the length of the actual variable name + 1 (for an ampersand). The symbol substitution service enforces this rule to prevent buffer overruns.

WAIT={ ESM | VTAM }

Specifies whether initialization waits for the External Security Manager (ESM) or VTAM to become available.

Dynamically Allocate a Data Set

The following command and its operands can be used to dynamically allocate a data set:

```
DD=name | DD=*
        [ ,DSN=datasetname ]
        [ ,DISP=( { OLD | SHR | MOD | NEW } [ ,ndisp ]
        [ ,cdisp ] ] ) ]
        [ ,VOL=volume ]
        [ ,UNIT=unitname ]
        [ ,{ CYL=( pri,sec ) | TRK=( pri,sec ) } ]
        [ ,BLKSIZE=nnnnn ]
```

If the operands cannot fit on one line, you can use the plus sign (+) to indicate that the command is continued. For example:

```
DD=FRED,+
        DSN=FRED.DATASET,+
        DISP=OLD
```

The operands are:

DD=*name* | DD=*

Specifies the DD name being allocated. This operand must be first and a valid DD name must be used. DD=* indicates that this data set is to be dynamically concatenated to the previous data set allocation (which must be a data set, not a SYSOUT allocation).

DSN=*datasetname*

Specifies the data set to be allocated. A PDS member name or GDG relative generation number can be specified in parenthesis after the data set name.

DISP=({ OLD | SHR | MOD | NEW } [,*ndisp* [,*cdisp*]])

Specifies the data set disposition. *ndisp* and *cdisp* are the standard UNCATLG, CATLG, DELETE, and KEEP options and default as in standard JCL. If only the first disposition is required, no parentheses are necessary. For example:

```
DISP=OLD
```

VOL=*volume*

Specifies the volume that the data set is (to be) on. For existing data sets, bypasses a catalog search. Generally requires that the UNIT operand also be specified.

UNIT=*unitname*

Specifies the unit type that the data set is (to be) on. This is required for existing data sets if the volume is specified.

CYL=(*pri*,*sec*) | TRK=(*pri*,*sec*)

For new data sets, allows specification of a space allocation value.

Note: You cannot allocate a PDS because no directory amount can be specified.

BLKSIZE=*nnnnn*

Lets you specify a block size. This is also useful when concatenating data sets of unlike block size because it sets the DCB BUFLEN value.

Dynamically Allocate a SYSOUT Data Set

The following command and operands may be used to dynamically allocate a SYSOUT data set:

```
DD=name
          SYSOUT=class
         [FREE={ CLOSE | END } ]
         [HOLD={ NO | YES } ]
         [BLKSIZE=nnnnn ]
```

If the operands will not fit on one line, an operand, you can use the plus sign (+) to indicate that the command is continued. For example:

```
DD=LOG1,+
         SYSOUT=A,+
         FREE=END
```

The operands to dynamically allocate a SYSOUT data set are:

DD=*name*

Specifies the DD name being allocated. Must be the first operand. A valid DD name must be used.

SYSOUT=*class*

Specifies the SYSOUT class desired. Use a single letter or number, A to Z, or 0 to 9. Specifying an asterisk implies that the job MSGCLASS is to be used (as in JCL).

FREE={ CLOSE | END }

CLOSE specifies that the SYSOUT is to be spun-off when closed, END specifies that it is not. Note that FREE=CLOSE is the default. You *should not* use FREE=CLOSE for FMTDUMP.

HOLD={ NO | YES }

Specifies if the SYSOUT is to be held or not. HOLD=NO is the default.

BLKSIZE=*nnnnn*

Lets you specify a block size.

Chapter 4: Tuning Performance

This section contains the following topics:

Enhance Performance (see page 35)
Tune at the System Level (see page 36)
Tune VTAM Interface (see page 36)
Tune Panel Use (see page 37)
Improve NCL Procedure Usage (see page 38)
Tune VSAM Data Sets (see page 39)
Tune Communication Between Regions (see page 46)
Control Message Flow in OCS (see page 49)
Performance Considerations When Writing NCL (see page 49)
Performance and Tuning Commands (see page 50)
Record CPU Usage (see page 61)

Enhance Performance

You can enhance the performance of your regions at each of the following levels:

- System
- Product
- Function

By following the performance enhancement measures outlined in this chapter, you will achieve the most effective use of your product.

Tune at the System Level

The first area to consider when tuning your region is at the system level. The following points outline system level performance considerations:

- For z/OS systems, the performance group and dispatching priority can affect the performance of your region. High-priority systems should be in the same domain as VTAM, and just below it in dispatching priority.
- In terms of tuning, your product can be regarded as equivalent to CICS or IMS. All system tuning rules that apply to other online systems should be used with these systems to ensure that your product region is not starved for real storage.
- Data set placement is important to consider when increasing the performance of your system. The PANELS, MODS, and NCL COMMANDS data sets can have a large amount of I/O activity on them. For this reason, it is wise to place these data sets on volumes and paths that are not otherwise busy. Ways of reducing the I/O activity on these data sets are described later in this chapter.
- Running your product region as non-swappable can help to increase response time. This is important when the overall activity rate may be low, but quick response is extremely important. The SYSPARMS NONSWAP=YES operand is used to specify whether your product region is non-swappable (the default).

Note: Some products must run non-swappable.

Tune VTAM Interface

To ensure that you gain maximum effectiveness from your products, the following performance points should be considered for your VTAM interface:

- Check any pacing specifications in System Services APPL statement parameters. They could affect INMC performance.
- If INMC is implemented using VTAM, dedicated COS definitions can be specified to optimize paths through the network (for multi-region environments only). In addition, load sharing can be achieved by grouping SOLVE:FTS initiators to different COS definitions.

Important! Do not use the Alias Name Translation facility unless you have to. Using this facility means taking the TR-INQ RU code X‘3F0814’ out of the CNM routing table, and therefore prevents VTAM from requesting redundant name translations.

Tune Panel Use

In some products, a large number of panels may need to be displayed. These panels are stored on VSAM data sets. Panels are retrieved from these data sets as required. Panel definitions are retained in storage for both reuse and automatic sharing.

Tune Storage Limits for Panel Sends

The maximum amount of storage allocated to allow panels to be sent to terminals is crucial to the performance of your region and is separate to considerations for tuning panel access. The following two SYSPARMS operands are used to specify storage amounts:

PANLBFSZ

Sets the amount of storage to be acquired to build a panel data stream. The default is 20 KB.

PANLBUFF

Sets the maximum number of pages of storage to be allowed for panel buffers

To avoid severe response time problems when you have a large number of application users, increase the value of the PANLBUFF operand. Otherwise, when a user requests a panel, they might have to wait until a previous panel send has been completed before they have access to their panel.

More information:

SYSPARMS Operands (see page 257)

Monitor Panel Send Storage Limits

To monitor how your system performs panel sends, use the SHOW GRP=PBUFPGT command. This command shows statistics about the panel buffer storage pool and can help you determine whether delays are being caused by panel send throttling.

The reason that the current storage utilization may be close to, or above, the defined maximum storage, is that some of the terminals that have had panels sent to them have not returned a definite response. This causes unnecessary throttling of panel send operations.

The system automatically increases the limit temporarily if the condition persists and returns the limit to its original value when the condition is relieved. This does not alter the short-term throttling characteristic, which is used to prevent flooding of the network and VTAM.

Note: You can alter the throttling characteristics dynamically by issuing the SYSPARMS PANLBUFF command to increase or decrease the active panel send limit. For example, to minimize the effect of panel sends not completing, you could increase the panel send limit after EASINET startup.

Improve NCL Procedure Usage

With some products or user-written applications, a concern may be the rate at which NCL procedures are loaded into storage. Modular programming practices encourage breaking an application into as many separate small modules as possible. This leads to many loading requests for procedures issued as NCL executes.

If this is not addressed, the following problems can occur:

- Applications run slowly, as they are constantly waiting for procedures to load. This is particularly evident in some applications that call a procedure to edit each field on a panel.
- I/O to the NCL procedure libraries can be a bottleneck.
- CPU time is expended in loading and pre-compiling the NCL.
- Real storage is tied up for I/O buffers.

Monitor NCL Procedure Loading Activity

To monitor the loading of NCL procedures for your system, use the SHOW NCLSTAT command. This command displays statistics about the number of:

- Load requests
- Actual loads
- Loads satisfied by preload
- Loads satisfied by autoshare and retain

By monitoring the increase in the value of these statistics across the day, you can use the following techniques to help strike a balance between loading activity and storage use:

- Sharing NCL procedures between users
- Preloading NCL procedures

Preload NCL Procedures

You can nominate a set of NCL procedures to remain preloaded by using the OCS command, LOAD. This involves a once-only load and precompilation. These procedures can then be shared by any number of users. Preloading NCL procedures is useful when you have short routines that are typically used by a single user (no concurrent usage). These procedures are constantly flushed from storage and reloaded again. It is also best to specify any heavily-used NCL procedures to be preloaded.

Note: For information about the LOAD command, see the online help.

Tune VSAM Data Sets

Most of these products rely on VSAM UDBs and NDBs. The way you manage your VSAM data sets has a direct effect on the performance of your products. The facilities described in the following sections allow you to tune UDBs and NDBs for maximum performance benefits.

Buffer Sharing

Buffer sharing is controlled by the VSAM LSR (Local Shared Resources) facility. By sharing buffers, data set I/O is reduced. The shared buffers are defined in LSR in pools.

To enable buffer sharing for a VSAM data set, use the UDBCTL OPEN command with the LSR operand.

Note: For files allocated through Customizer parameter groups, you can enter UDBCTL operands as VSAM options. For more information about the UDBCTL command, see the online help.

When specifying the LSR pool definition that best suits your needs, consider the following factors:

- Your need for virtual storage versus I/O performance
- The mix of VSAM files that you have open concurrently in the address space and which of those you want to place in the LSR pool

To define your LSR pool definitions, use the LSRPOOL parameter group (enter **/PARMS**).

Note: VSAM alerts (see page 43) warn about string or buffer shortages.

Deferred Write Capabilities

Deferred write capabilities let you defer the updating of data sets so there are fewer I/O requests. The advantages of using this facility should be weighed against the possibility of losing data in the event of a system failure.

To allow a data set to have deferred write capabilities, specify the DEFER operand on the UDBCTL OPEN command, or VSAM options in a Customizer parameter group.

For more information about the UDBCTL command, see the online help.

VSAM Processing in a Subtask

You can nominate VSAM I/O to be performed by a subtask rather than the System Services main task. This provides increased throughput by overlapping VSAM processing. To nominate a subtask to perform VSAM I/O, use the VSAMIO JCL parameter.

Note: Using a subtask is only useful when there is significant VSAM processing and you have a multi-CPU machine.

You can use the SHOW VSAMIO command to obtain statistics of VSAM processing.

Example

Following is an example of SHOW VSAMIO command output:

```
(18.51)---------------- Operator Console Services (OCS) -----------------------
show vsamio
N13A10 VSAM I/O MANAGER STATISTICS.
N13A11 MODE: DYNAMIC MOLAP: 20 MBPWAIT: 5 DSTS: 2 DSTM: 0 REQS: 173K
N13A12 TASK REQUESTS COL POL TIMES-POL NOWAIT PSWAIT PENDING
N13A13 MAIN 133K 0 2 1229 111K 11107 -
N13A13 SUB 39491 0 5 1 37542 - 0
N13A14 OLAP M/T S/T
N13A15 0 0 7300
N13A15 1 21145 1448
N13A15 2 1229 237
N13A15 3 0 22
N13A15 4 0 2
N13A15 5 0 1
N11907 *END*

==> show vsamio
```

More information:

Product Region JCL Parameters (see page 247)

Record Size Management

When tuning the size of your databases, you need to consider the control interval size (CISIZE), maximum record size (RECSZ), and free space.

NDB database records should be tuned to fit into a single VSAM record.

Note: For information about the storage needed by an NDB record, see the *Network Control Language Programmer Guide*. Use this information to choose a suitable CISIZE that gives good track utilization.

If the NDB is subject to heavy update activity, pick a reasonably large free space amount. This enables data and key record alterations in the NDB structure to take place without causing excessive numbers of CI/CA splits.

You should also regularly reorganize your databases using IBM's IDCAMS REPRO command.

Note: Currently, you cannot change the record size of an NDB without logically unloading and reloading it.

Exceptions

The following NDBs store historical records:

- Alert history ($ALERTH)
- File transfer events log (EVENTDB)

Because records are added and deleted in key sequence order, these NDBs should not have free space allocated. Also, they should not be reorganized because the space used by deleted records is reclaimed by the NDB feature, RIDREUSE, which is active for these NDBs. You can monitor their sizes by defining VSAM alerts (see page 43) that are raised when a predetermined number of extents are allocated.

Database Activity Management

Database activity tuning applies to NDBs. Consider the following points when using NDBs to enhance performance:

- Open an NDB as a UDB by using the UDBCTL OPEN command with the LSR and DEFER options.
- Define the LSR pool with sufficient buffers of appropriate size for the NDB data set data and index.
- Check that SYSPARMS, NDBSUBMN, and NDBSUBMX values are adequate for the amount of concurrent SCAN activity that is occurring.
- Choose values for the NDB scan limit system parameters that stop users from running scans that consume excessive system resources.

Important! Do not use the DEFER option of the NDB START command unless you are bulk loading.

Customize VSAM Alerts

An alert is always raised when a file is full or an error condition occurs; however, you can modify the severity. You can also specify whether to raise an alert for VSAM file extensions, string and buffer shortages, and enable alerts when a UDB expands to a pre-defined number of extents. You can customize these conditions through the VSAMMONITOR Customizer parameter group.

To customize VSAM alerts

1. Enter **/PARMS** at the prompt.

   The Customizer : Parameter Groups panel appears.

2. Enter **U** beside VSAMMONITOR in the Monitors category.

   The VSAMMONITOR Customizer panel appears.

3. Complete the fields.

   Note: For information about the fields, see the online help.

   a. (Optional) If you want to apply the changes immediately, press F6 (Action).

      The region takes on the changes.

   b. Press F3 (File).

      The changes are saved.

Performance Monitoring Facilities

The SHOW VSAM command is provided to display attributes and statistics about VSAM databases.

Example

Following is an example of a SHOW VSAM display.

```
(02.25)---------------- Operator Console Services (OCS) -----------------------
show vsam
N15101 DDNAME   RECSZ D-CISZ I-CISZ CI-SP CA-SP D-BF I-BF STRSH BFRSH LSR CTL
N13522 VFS       4089   8192   2048    75     0   10    9     0     0 NO  DSN
N13522 USERIDS   4000   4096   2048     0     0    4    5     0     0 NO  DSN
N13522 OSCNTL   32700  32768   4096    41    10    4    3     0     0 NO  DSN
N13522 NMLOG01   4089  22528   2048     2     0    0    0     0     0 YES DSN
N13522 NMLOG02   4089  22528   2048     0     0    0    0     0     0 YES DSN
N13522 NMLOG03   4089  22528   2048     0     0    0    0     0     0 YES DSN
N13522 MODSUSR   4096   8192   2048     0     0    0    0     0     0 YES DSN
N13522 MODSDIS   4096  16384   2048  1213    97    0    0     0     0 YES DSN
N13522 PANLUSR   8185   8192   2048     0     0    3    3     0     0 NO  DSN
N13522 PANLDIS   8185   8192   2048    70     9    3    3     0     0 NO  DSN
N13522 ICOPANL  16377  16384   2048     0     0    3    3     0     0 NO  DSN
N13522 ALERTH    8185  16384   4096     0     0    0    0     0     0 YES DSN
N13522 NETINF1   2048   4096   2048     0     0    0    0     0     0 YES DSN
N13522 PSPOOL     340   4096   4096     0     0    0    0     0     0 YES DSN
N13522 RAMDB     8185   8192   4096     6     0    0    0     0     0 YES DSN
N13522 RAMDBST    200   8192   4096     0     0    0    0     0     0 YES DSN
N13522 IPFILE    8185   8192   4096     0     0    0    0     0     0 YES DSN
N13522 IPMIBX    4089   4096   2048     0     0    0    0     0     0 YES DSN
N13522 IPLOG     4089   4096   4096     2     0    0    0     0     0  11 DSN

==> SHOW VSAM
```

Monitor LSR Tuning

To assist with specifying the LSR pool definitions that provide the best performance for your system, use the SHOW LSR command. This command displays statistics about the LSR pool. From this display you can note trends and alter the LSR pool definition appropriately.

Example

The following panel displays LSR pool statistics from a system that has been customized, and as a result has significantly more buffers than the distributed definition.

```
(23.01)---------------- Operator Console Services (OCS) ----------------------
SHOW LSR
N15A30 ACTIVE LSR POOL 0 STATISTICS
N15A31 KEYLEN: 255 STRNO: 220 FIXIOB: NO FIXBFR: NO STRMAX: 7 ACTIVE: 19
N15A32   SIZE COUNT  P. READS BUF FOUND       UIW      NUIW %FOUND  HS-COUNT
N15A33   2048   100     80758   1249419    169112         0  93.92
N15A33   4096   200       197    951843      6061         0  99.97
N15A33   8192    30      7346    393443      1884         0  98.16
N15A33  10240    20         0         0         0         0   0.00
N15A33  16384    20         0         0         0         0   0.00
N15A33  28672     6        22     27066       338         6  99.91
N15A30 ACTIVE LSR POOL 1 STATISTICS
N15A31 KEYLEN: 255 STRNO: 250 FIXIOB: NO FIXBFR: NO STRMAX: 1 ACTIVE: 1
N15A32   SIZE COUNT  P. READS BUF FOUND       UIW      NUIW %FOUND  HS-COUNT
N15A34 DATA...
N15A33   8192   500       413     42172       513         0  99.03        50
N15A35 INDEX...
N15A33   2048   100        11     85045         1         0  99.98
N15A30 ACTIVE LSR POOL 11 STATISTICS
N15A31 KEYLEN: 255 STRNO: 12 FIXIOB: NO FIXBFR: NO STRMAX: 1 ACTIVE: 1
N15A32   SIZE COUNT  P. READS BUF FOUND       UIW      NUIW %FOUND  HS-COUNT
N15A33   2048    40         6     15688        40         0  99.96

==> SHOW LSR
```

In this example, little use is being made of the 512-byte buffers. The sum of pool reads plus the number of buffers found is a small number. This small number is a sign that there are too many buffers in the pool. Deleting the pool might also be an option but could also be quite wasteful of storage as buffers from the next larger pool will then be used.

There is a low percentage of 12-KB and 16-KB pools found. This is an indication that the number of buffers could be increased for those pool sizes.

Monitor Data Set Activity Tuning

Use the SHOW NDB=ALL command to monitor the amount of concurrent SCAN activity that is occurring within your system. Then use the SYSPARMS, NDBSUBMN, and NDBSUBMX operands to tune activity appropriately.

Example

The following is an example of SHOW NDB=ALL command output.

```
(20.56)---------------- Operator Console Services (OCS) -----------------------
SHOW NDB=ALL
N89503 NAME     STATUS     QCMD  USERS CMDS-DONE SUBT QSUB DFR LSIZ LHWM TSILJR
N89504 RAMDB    ACTIVE        0      2       821    3    0 NO    40    4 NNNNNP
N89504 $ALERTH  ACTIVE        0      0         5    0    0 NO    40    1 NNNNNP
N89505 *END*

_______________________________________________________________________________

==> SHOW NDB=ALL
```

Tune Communication Between Regions

Many sites make heavy use of INMC. By ensuring that your INMC links are running smoothly you can improve the performance of the facilities that rely heavily on INMC. The following features can be tuned to aid in the smooth communication between your domains using INMC:

- INMC
- ROF
- FTS
- ISR

Tune INMC

There are two areas in which INMC tuning can be achieved:

- INMC buffer size—specified by the SYSPARMS INMCBFSZ operand
- BIND RUSIZE—specified in the logmode table definition

If you increase the INMCBFSZ value, and the RUSIZE value is not at least this large, no benefits will be derived.

INMC also lets logmodes be used with pacing turned on. This can prevent a VTAM link being overrun by INMC traffic.

More information:

SYSPARMS Operands (see page 257)

Tune ROF

To prevent excessive message traffic being sent across INMC links during ROF connections, the following techniques can be used:

- Always configure the remote user ID to receive only relevant message traffic.
- Avoid duplication of messages. Operators should only receive unsolicited messages in remote domains if they really need them.
- Consider signing on background regions across a ROF link. For example, if BMON is signed on to a remote domain, any messages it is eligible to receive from the remote domain may be processed by a MSGPROC in the receiving system, and then propagated to all MONITOR receivers in that system. Only one copy of the message flows across the link in this case.

  In this case operators that need to issue commands to the remote domain do not need to be profiled to receive unsolicited message receipt.
- Consider using Inter-System Routing (ISR) facilities if applicable.
- Use the SIGNOFF command to terminate remote connections and free resources.

Tune CA SOLVE:FTS

Note: The section applies only to CA SOLVE:FTS.

The performance of CA SOLVE:FTS is influenced by the following external factors:

- The block sizes of the transmitted data sets
- INMC tuning
- The speed at which data can be transmitted. Consider the INMC feature which can make use of multiple parallel paths between two domains. For example, create INMC-specific logmodes which have COS definitions that traverse different links at different priorities.
- Priority of File Transmission Services traffic. File Transmission Services has the lowest INMC priority. This is not adjustable.

The only internal tuning that can be performed in CA SOLVE:FTS is to limit the number of initiators for a specific destination to prevent excessive resource commitment. This can be done by the CA SOLVE:FTS initiator function or through the FTSINIT command.

Note: For more information about the FTSINIT command, see the online help.

Tune ISR

When using ISR, carefully consider the types of data that are to be sent across an INMC link. To monitor ISR data exchange, use the SHOW ISRSTATS command to obtain statistics.

Control Message Flow in OCS

To ensure that OCS activity impedes the performance of your systems as little as possible, the following points should be considered:

- Restrict the types of messages that an operator receives using UAMS or an external security definition.
- Do not allow all operators to receive unsolicited messages by default.
- Use MSGPROCs. Held message limits can be reached while the panels are being displayed. If messages are not going to be examined, they should not be sent. These messages waste storage space and CPU time.
- Restrict the use of non-roll delete (NRD) messages. Extra system resources are needed to retain NRD messages.
- Encourage the use of selective commands. Monitor the use of $CMDENT. If excessive, perhaps the default wrap count should be lowered.
- Use the SYSPARMS HELDMSG=*xxx*,*yyy* command to limit the hold message queue. The queuing of messages causes excessive storage usage and fragmentation.

Performance Considerations When Writing NCL

When writing NCL, the following points should be considered to ensure that your system performance does not suffer:

- Ensure that the &CONTROL RESCAN option is only used around statements to which it applies. Rescanning every NCL statement slows the system down.
- Use suppressed comments because they occupy no storage. They add an insignificant overhead to the procedure load process only.
- Use modular programming.
- Use the NCLTEST command when testing NCL to circumvent the retain and autoshare facility.
- Do not use static and dynamic PREPARSE facilities for panels unless they are essential to a specific panel display.
- In any NCL procedures that display panels, think about the number of variables that are active across any panel displays. The large number of concurrent processes executing can result in excessive virtual storage usage. Comment out any unneeded variables before any of these displays.
- Reduce the overheads of using &CALL repeatedly to call a module by using one of the following methods:
  - Use the LOAD MOD=*module_name* command, if the module is reentrant, to load it once into storage.
  - Use the SUBSYS facility to attach the program once.

More information:

Improve NCL Procedure Usage (see page 38)

Use Commands to Control Resource Consumption by NCL

There are several commands that can be used to alter the performance of your region. These system performance commands include SYSPGT (see page 58), SYSRCT (see page 58), SHOW SYSWAIT (see page 56), SHOW SYSPGT (see page 51), and SHOW SYSRCT (see page 53). These commands display and control resource consumption by NCL processes.

All NCL statements are assigned a processing unit weighting. SYSRCT and SYSPGT commands let you set up performance groups whereby an NCL process is placed into a forced wait if it consumes a certain amount of these processing units. This allows System Services to run with a high priority in the operating system, but prevents them from using all available CPU time if an NCL procedure starts looping.

System-level procedures should have a high priority. Short-term performance controls can be used to stop a runaway system procedure without impacting it in the long term.

The SHOW NCL command can also be used to indicate the performance group, priority, and current processing unit consumption. For more information about the SHOW NCL command, see the online help.

Important! Busy system procedures that do not wait because there is always a message available for &xxxREAD may be unfairly penalized by the short-term performance control. These should be reset using the following command:

```
SYSRCT G=4 P=0 SDELAY=(0,0,0,0)
```

Performance and Tuning Commands

The following commands set the performance parameters of the different categories of process that your region executes. The commands also display the relative activity in a region.

The default settings are designed to cater for most installation requirements, but these commands are available to change the defaults if necessary.

Important! These commands should be used with caution by skilled system administrators.

SHOW SYSPGT

The SHOW SYSPGT command displays the summary information about the NCL system performance group tables.

The system performance group table contains historical statistics about each of the four performance groups. The table also contains the priority that is given to NCL procedures when they start executing under the control of a particular group. The statistics show the number of currently active procedures, the total started and totals of performance control actions taken for each group.

This command has the following format:

```
SHOW SYSPGT
(0)
```

Example

```
SHOW SYSPGT
N58013 PG--IPRTY-ACTV-UPDATED---STARTED---DELAYS----CPRTY
N58004  1    1      3 14.42.05       13        0        0 BACKGROUND
N58004  2    1      2 14.42.05      121       17        3 OCS
N58004  3    2      0 14.42.05        3        0        0 FULLSCREEN
N58004  4    1     57 14.42.05     2288       54       24 SYSTEM
N13503 *END*
```

Return Information

PG

The performance group number.

IPRTY

The priority that is assigned to a procedure when it starts executing.

ACTV

The number of currently active procedures.

UPDATED

The time RCT control values were set. Initially, these values are set during system start-up.

STARTED

The total number of procedures that have run in this performance group.

DELAYS

The total number of forced waits that have occurred for all procedures that have run in this performance group.

CPRTY

The total number of priority changes that have occurred for all procedures that have run in this performance group.

text

The descriptive name of the performance group.

Performance Groups

BACKGROUND

These are procedures that execute without a physical terminal environment: BGMON and BGLOG.

OCS

These are line mode procedures executing in association with a real OCS window, including commands from ROF sessions.

FULLSCREEN

These are procedures which use panel services, including EASINET.

SYSTEM

These are system-level procedures: BSYS, PPOPROC, CNMPROC, LOGPROC, AOMPROC, RMINIT, and RMREADY procedures, and CA SOLVE:FTS commands.

SHOW SYSRCT

This SHOW SYSRCT command displays summary or detail information about NCL performance groups. It can be used to obtain either summary information about all four NCL performance groups, or detailed statistics about a specific NCL performance group.

This command has the following format:

```
SHOW SYSRCT
(0)      [ =SUMMARY | =n  ]
```

=SUMMARY | =*n*

The default (=SUMMARY) displays a single line for each performance group.

If =*n* is specified, detailed information about the nominated performance group is shown, with all priority levels displayed.

Example

```
SHOW SYSRCT=4
N58007 PERFORMANCE GROUP 4, SYSTEM , 1430 STARTED SINCE - 14.42.05
N58008 PRIORITY+--TRIGGERS-+-------CONTROLS-------+--------STATISTICS-------+
N58009   3      DELAY CPRTY INIT. ADJ. LIMIT NPRTY DELAYS  LIMIT  CPRTY   ACT
N58010 SHORT      -     -     0    0     0    3        -      -      -     0
N58011 LONG       -     -     0    0     0    3        -      -      -
N58009   2
N58010 SHORT      -     -     0    0     0    2        -      -      -     0
N58011 LONG       -     -     0    0     0    2        -      -      -
N58009   1 <=I=
N58010 SHORT     20    10    50    0    50    0        0      0      0    54
N58011 LONG       -    80     0    0     0    0        -      -     13
N58009   0
N58010 SHORT     20     -    50    0    50    0        0      0      -     3
N58011 LONG       -    80     0    0     0    1        -      -     10
N13503 *END*
```

Return Information

PG

The performance group number

ACTV

The number of currently active procedures.

INTERVAL

The statistics recording start time.

STARTED

The total number of procedures that have run in this performance group.

DELAYS

The total number of forced waits that have occurred for all procedures that have run in this performance group.

LIMIT

The total number of times that a forced wait has been issued for the maximum or minimum duration.

CPRTY

The total number of priority changes that have occurred for all procedures that have run in this performance group.

Detailed Information

The detailed information returned includes the previous statistics for long-term and short-term performance control measures, and the following performance control settings for priorities and forcing procedures to wait:

DELAY

The number of processing units that can be consumed by a procedure before a forced wait occurs.

CPRTY

The number of processing units that can be consumed by a procedure before a priority change occurs.

INIT

The length of time in hundredths of a second that a procedure is delayed for, after consuming the DELAY number of processing units for the first time.

ADJ

The length of time in hundredths of a second that the initial DELAY is adjusted by for subsequent times that the delay number of processing units is consumed.

LIMIT

The maximum or minimum length of time a procedure is forced to wait.

NPRTY

The new priority that is assigned to a procedure when it has consumed the CPRTY number of processing units.

The Resource Control Table contains controls for dynamically altering the performance of currently active NCL procedures based on their consumption of *processing units* and statistics recording their effects.

Each priority in a performance group has short term and long-term controls. Short-term controls apply to processing units consumed since the last voluntary loss of control, for example, an &PANEL statement. Long-term controls apply to processing units consumed since the procedure was initiated.

The four performance groups are BACKGROUND, OCS, FULLSCREEN, and SYSTEM, numbered 1 through 4 respectively.

Each performance group has an initial priority which can be displayed using the SHOW SYSPGT command. In the detailed display of the resource controls for a particular group (that is, SHOW SYSRCT=*n*), the initial priority is indicated by <=I=.

SHOW SYSWAIT

The SHOW SYSWAIT command displays main task apparent wait statistics.

The system maintains wait statistics each time an operating system WAIT is issued. The information returned by this command includes the following statistics:

- Percentage of elapsed time that your product region is in an operating system wait (AWAIT%)
- Percentage of times a wait occurred when an NCL procedure was forced to wait due to performance controls (FWAIT%).

These statistics can be used as a guide to how busy the system is and whether performance control measures are having an overall effect.

This command has the following format:

```
SHOW SYSWAIT
(0)
```

Example

```
SHOW SYSWAIT
N58001 PERIOD START:   AWAITS   FWAITS   FWAIT%   AWAIT%     RATIO
N58002   14.30.04      135385     3558    2.62%   99.36%         2
N58002   14.30.04      135385     3558    2.62%   99.36%         2
N58020 PERIOD START:    TOTAL-CPU %-BSY
N58021                 SYSTEM-CPU %-SYS  ACCNTED-CPU %-ACC UNACCNTD-CPU %-UNA
N58022   14.30.04      140.173343   0.6
N58023                   3.118633   2.2   137.047259  97.7     0.007450   0.0
N58022   14.30.04      140.173343   0.6
N58023                   3.118633   2.2   137.047259  97.7     0.007450   0.0
N13503 *END*
```

Return Information

START

The start time for statistics accumulation.

AWAITS

The number of operating system waits that have occurred since the start time.

FWAITS

The number of FWAITS that took place when an NCL procedure was in a forced wait due to performance control measures.

FWAIT%

The percentage ratio FWAITS is to AWAITS.

AWAIT%

The percentage of elapsed time that the system was in an operating system wait.

RATIO

The number of interval events processed per AWAIT.

Use SYSWAIT statistics as a guide only because they reflect the voluntary loss of control of your product region and do not reflect operating system resource management activity.

The statistics are maintained on a historical and interval basis. The first line of the two detail lines shows the statistics since this command was last issued. The second detail line shows the long-term accumulation. These statistics are reset at midnight. Each time this command is issued, the current statistics for this interval are rolled over into the historical statistics. This feature provides a means for displaying activity in the short term, compared against the long-term average.

AWAIT% reflects the apparent wait time that control has voluntarily been passed to the operating system. Therefore, the actual wait time could be higher, because of page faults, higher priority work, and so on.

The number of times that at least one NCL process was forced idle when a wait was issued is also shown as the FWAIT count. This count indicates that work could have been done, but performance controls forced all ready NCL processes to wait.

SYSPGT

The SYSPGT command is used to set the initial priority for an NCL performance group. It is used as part of performance control. It sets the priority that is assigned initially to procedures running in a performance group.

This command has the following format:

```
SYSPGT   GROUP=n
         IPRTY=p
(4)
```

GROUP=*n*

This is the number of the performance group that is to have its initial priority set. The valid range is 1 to 4.

IPRTY=*p*

This is the initial priority for the specified performance group. The valid range is 0 to 3, zero being the lowest priority.

Examples

```
SYSPGT G=1 IP=1
SYSPGT GROUP=4 IPRTY=3
SYSPGT GR=2 I=2
```

The default initial priority for all performance groups can be determined using the SHOW SYSPGT command.

The four performance groups are BACKGROUND, OCS, FULLSCREEN and SYSTEM, numbered 1 to 4, respectively.

A procedure can have its priority altered during execution depending on control values set in the Resource Control Table. See the description of the SYSRCT command for details.

SYSRCT

The SYSRCT command is used to set performance control parameters in the system Resource Control Table (RCT). It is used for performance control. It can be used to set *processing unit* trigger values and performance control parameters. The following types of performance control measure are supported:

- A priority change control measure
- A forced wait control measure

Performance is calculated in the following ways:

- Short-term—that is, between voluntary waits, for example, screen interaction. The SDELAY and SNPRTY operands are used for short-term evaluation.
- Long-term—that is, for the *life* of the NCL process. The LDELAY and LNPRTY operands are used for long-term evaluation.

This command has the following format:

```
SYSRCT  GROUP=n
        PRIORITY=p
        [ LDELAY=(  [ trigger ], [ initial ], [ adjustment ], [ limit ] )  ]
        [ SDELAY=(  [ trigger ], [ initial ], [ adjustment ], [ limit ] )  ]
        [ LNPRTY=(  [ trigger ], [ new-priority ] )  ]
        [ SNPRTY=(  [ trigger ], [ new-priority ] )  ]
(4)
```

GROUP=*n*

The number of the performance group (range 1 to 4) that is to have its RCT values updated.

PRIORITY=*p*

The priority (range 0 to 3, zero being the lowest) that is to be updated.

LDELAY=([*trigger*], [*initial*], [*adjustment*], [*limit*])

SDELAY=([*trigger*], [*initial*], [*adjustment*], [*limit*])

These operands are used to set values for forcing procedures to wait. The trigger value specifies the number of 'processing units' that a procedure can consume before a wait is forced. The three parameters initial, adjustment and limit, specify times in hundredths of a second with a maximum absolute value of 200 (that is 2 seconds).

The initial value is used for setting the period of the first wait. Subsequent wait times might be longer or shorter, being incremented or decremented by the adjustment value, which can be negative or zero. Consequently, the limit can be a maximum or minimum wait time. A zero adjustment results in the limit being set to the initial value, overriding any existing or specified value. When no limit value has been set and a positive adjustment has been specified a default maximum limit of 200 is used.

LNPRTY=([*trigger*], [*new-priority*])

SNPRTY=([*trigger*], [*new-priority*])

These operands are used to set values for altering the priorities of procedures as they execute. The trigger value specifies a number of processing units that can be consumed before a priority change occurs. The new priority must be 0 to 3 and can be the same as the current priority.

Examples

```
SYSRCT G=1 P=1 LNP=(100,2)
SYSRCT GROUP=1 PRIORITY=2 SDELAY=(20,5,1,20)
SYSRCT GR=1 PR1=2 SDELAY=(,,,30)
```

The first example sets a long-term priority change for procedures in group 1 (BACKGROUND). Priority 1 is to be altered to priority 2 after the consumption of 100 *processing units*. The SYSPGT command can have been used to set the initial priority for this performance group to be 1.

The second example sets a forced wait for procedures running in priority 2 to occur every 20 *processing units*. The first wait would be for an interval of 5 hundredths of a second, the second would be 6, the third 7, and so on, until the maximum of 20 hundredths (0.2 seconds) is reached.

The third example alters only the short term delay value limit to be 30 hundredths of a second.

Processing units are a simple measure of work arbitrarily assigned to the execution of NCL statements. They are in no way an accurate measure of actual work performed, or an approximation of CPU time consumed. You cannot compare the work done by two different NCL processes based on processing units consumed.

There are two sets of processing unit consumption statistics for each NCL procedure—one for short-term evaluation and the second for long-term evaluation. It also maintains a long-term dispatching priority as well as a current dispatching priority. Short-term statistics are reset whenever a voluntary loss of control occurs or a priority change takes place. A short-term priority change alters only the current dispatching priority, whereas a long term priority change alters both priorities.

The order of evaluation of performance control measures is as follows:

1. Long-term forced wait and priority change check
2. Short-term forced wait and priority change check

Only the first of the above that is triggered is actioned; however, both a priority change and a wait can occur at the same time, with the wait being actioned before the priority change. Procedures started using an &INTCMD statement are run in the owning process's performance group, starting at the performance group's initial priority.

The four performance groups are BACKGROUND, OCS, FULLSCREEN, and SYSTEM, numbered 1 to 4 respectively.

Record CPU Usage

This section describes how to use the CPU-time accounting facilities of NCL.

NCL CPU-time Accounting

NCL CPU-time accounting lets you collect NCL CPU usage data at user level. This data can then be used to generate reports on the CPU usage of each user.

NCL CPU usage data is provided by the generation of System Management Facility (SMF) records at regular time intervals. These records contain the user ID, and the amount of CPU-time used by that user, including multiple signons, and background APPC or ROF regions.

You can control the time interval at which records are generated, and the CPU-time threshold at which a record is generated using the USERACCT (see page 63) command.

You can also display the current status of CPU-time accounting (see page 65).

Implement NCL CPU-time Accounting

To implement NCL CPU-time accounting, the OPT=01 JCL parameter must be specified and SMF record generation must be enabled by the SMF parameter group. Your system must also be running *authorized* for SMF reporting to occur.

Note: CPU-time accounting might impede the performance of your system.

Generation of SMF Records

SMF records are generated under the following circumstances:

- If the amount of CPU-time used by the user since the last record was generated, surpasses a predefined threshold
- On system initialization if user NCL CPU-time accounting is active at the time of shutdown

Whenever an SMF record is generated for a user, their CPU-time is reset to zero. Each SMF record contains the amount of CPU-time used by the user since the previous SMF record was generated.

Adjust Threshold Levels and Report Intervals

You can generate records for all of the users by changing the threshold value to zero. This gives you a cutoff value for CPU usage at the end of the day.

Creating reports at intervals during the day is useful if the system fails. There will still be CPU usage information available.

Setting a minimum threshold level obtains update information for the heavy CPU users. Time is not wasted generating incremental reports for users who use little or no CPU time.

SMF Record Format

The SMF record format consists of the following sections:

- Standard SMF record header
- User ID
- CPU-time

The format is shown in the following table.

Offsets	Length	Format	Source	Description
Standard SMF Header				
0	2	binary	internal	Record length
2	2	binary	internal	Segment descriptor (0000 as record not spanned)
4	1	binary	SVC 83	System indicator
5	1	binary	user supplied	Record type
6	4	binary	SVC 83	The time since midnight in hundredths of a second that record was moved to SMF buffer
10	4	packed	SVC 83	The date the record was moved to the SMF buffer, in the form 00YYDDDF (where F is the sign)
14	4	character	SMCASID	System identification (taken from the SID parameter)
Prefix				
18	1	binary	internal	Sub-category: X'06' for user CPU-time accounting
19	1	null	-	Reserved
20	12	character	internal	NMID of product region

Offsets	Length	Format	Source	Description
CPU-time Accounting Section				
32	8	character	internal	User ID to which the CPU-time value applies
40	1	binary	internal	Code indicating where CPU-time was used (X'00' is NCL CPU-time)
41	1	null	-	Reserved
42	8	stck	internal	CPU-time accumulated by user since last SMF write (stck format, where bit 51=1 microsecond)
50	4	binary	SVC 83	The time since midnight, in hundredths of a second, that the last record was cut for this user (0 if first record)
54	4	packed	SVC 83	The date the last record was cut for this user, in the form 00YYDDDF (where F is the sign) (null if first record)

Start and Stop CPU-time Accounting

The USERACCT command is used to start and stop NCL CPU-time accounting. It is also used to set the SMF record ID, and parameters controlling the timing and level of SMF reporting. Some of these parameters affect the time interval and start time of SMF report generation, so they can only be set if NCL CPU-time reporting status is inactive.

USERACCT Command

The USERACCT command has the following format:

```
USERACCT  START
          [ SMFRECID=smf-record-id ]
          [ INTERVAL=hours-between-smf-records
            [ FROM=start-time-for-first-report ]
            [ MINCPU=minimum-CPU-time-to-report ] ]

USERACCT  SET
          [ SMFRECID=smf-record-id ]
          [ INTERVAL=hours-between-smf-records
            [ FROM=start-time-for-first-report ]
            [ MINCPU=minimum-CPU-time-to-report ] ]

USERACCT  STOP
(4)
```

START

Activates NCL CPU-time reporting. This operand controls whether SMF reports on the statistics are generated. The actual accumulation of CPU-time statistics occurs independently of the reporting process, thus CPU-time statistics are collected irrespective of whether reporting is active or inactive.

This operand is accepted only if reporting is currently inactive.

SMFRECID=smf-*record-id*

Specifies the SMF record ID to be used on SMF records that contain NCL accounting information. It is effective immediately after the command is issued, so the next record generated uses this smf-record-id value.

Limits: 128 to 255.

Default: The default SMF record ID. This is specified in the SMF parameter group (enter **/PARMS**). If there is no default SMF record ID and this operand is not specified, the reporting thread fails with an invalid record ID message.

INTERVAL=*hours-between-smf-records*

Specifies the number of hours between the generations of SMF records. This operand is only allowed if NCL CPU-time reporting is inactive at the time the command is entered.

Limits: 0 to 24 (note, 0 is equal to 24). Only numbers which are whole factors of 24 (that is, 1, 2, 3, 4, 6, 12, and 24) are valid.

Default: 24

FROM=*start-time-for-first-report*

Specifies the base time for the calculation of intervals. The default if this operand is not entered is 00.00.00 (midnight). The time that the first SMF record is generated is calculated by adding the hours-between-smf-records value to the start-time-for-first-report value.

For example, if the hours-between-smf-records value is 4 hours, and the start-time-for-first-report value is 01.10.00 (1.10 am), and the command is entered at 08.00.00 (8.00 am), then the next report will be at 09.10.00 (9.10 am). This is because, using the base time and interval, record generation occurs at 1.10, 5.10, 9.10, 13.10, 17.10, 21.10, and 1.10 again. So, if the command is entered at 8.00 am, the next report is due at 9.10 am.

MINCPU=*minimum-CPU-time-to-report*

Specifies the minimum amount of CPU-time that must have been accumulated since a report was last generated for a user, before another report is generated for that user. It is effective immediately after the command is issued, and is applied on the next report.

Limits: A value in CPU seconds, with up to 2 decimal places (for example, 1.25 seconds).

Default: 0

SET

Modifies the NCL CPU-time reporting parameters. It can be specified whether CPU-time reporting is active or inactive; however, the FROM and INTERVAL operands are accepted only when CPU-time reporting is inactive.

STOP

Stops user NCL CPU-time reporting. This operand controls whether SMF reports on the statistics are generated. The actual accumulation of user CPU-time statistics occurs independently of the reporting process, thus CPU-time statistics are collected irrespective of whether reporting is active or inactive.

This operand is accepted only if reporting is currently active. No other operands are valid in conjunction with STOP.

Note: NCL CPU-time accounting occurs independently of the report generation process. A CPU-time report for a user contains a field which shows the time at which the last report for that user was generated.

Display the Status of NCL CPU-time Accounting

The SHOW USERACCT command displays the current status of user NCL CPU-time accounting.

The information displayed shows the following:

- Whether NCL CPU-time accounting is active
- The SMF record ID
- The start time for reporting
- The SMF reporting interval
- The CPU-time threshold for reporting

The start time is in the form *hh.mm.ss*, the interval is in hours, and the threshold is in hundredths of a second.

Chapter 5: Using the SYSCMD Facility

Note: This chapter does not apply to CA SOLVE:FTS.

This section contains the following topics:

SYSCMD Facility (see page 67)
Use SYSCMD in an &INTCMD Environment (see page 70)
Prefix Messages (see page 71)
Facilitate User-Generated Messages (see page 71)
SYSCMD Facility Commands (see page 75)
SYSCMD NCL Verb and System Variable Summary (see page 75)

SYSCMD Facility

The SYSCMD facility lets you issue z/OS commands from your region.

Set Console Time-outs

A console that is assigned to an environment may be taken by another user if that console has not been used for a specific period of time. The only time that this cannot occur is when a console is locked to an environment, or if a solicited WTOR has been received by that environment.

More information:

SYSPARMS Operands (see page 257)

Lock Consoles to a Specific Environment

In most cases, the pool of JES consoles can be shared by any number of users on a temporary basis.

However, there are cases where a specific environment must not miss out on obtaining a console, for example, if the pool of consoles is temporarily exhausted, or the consoles are timed out and are unable to be reassigned.

Note: We recommend that you avoid using JES consoles and use Extended MCS consoles (XMCS) instead.

You can lock consoles using the SYSCMD OPT=LOCK command.

Important! If you use SYSCMD OPT=LOCK from an NCL procedure and the procedure needs to issue an &INTCLEAR, then make sure you use &INTCLEAR TYPE=ANY rather than &INTCLEAR TYPE=ALL. &INTCLEAR TYPE=ALL will release the locked console.

Clear a Lock

When locked, the console can be released only when one of the following events occurs:

- The console owner issues the SYSCMD OPT=REL command.

  Note: Although this releases the console from the owner, the console cannot be used by anyone else until the relevant timeout periods have elapsed.
- The OCS window is exited, terminating the dependent processing environment.
- You use the AOM STOP command to stop SYSCMD processing.

Simulate Command Issue from a Specific Console

If you want to simulate issuing a command from a specific console that you are not using, enter the following form of the SYSCMD command (where *n* is the console number):

```
SYSCMD CON=n DATA=command-text
```

This can be done from anywhere in the system.

Because no console authority checks are made by the SYSCMD facility, the existing authority for the specified console is used.

Note: The form of the SYSCMD command described in the example above is not supported if you are using EXTMCS consoles.

The SYSCMD CON=*n* DATA=*command-text* form of the SYSCMD command is useful in NCL procedures started under console user IDs.

Example

A console operator, on Console 15, issues the following z/OS modify command to send a command to your region:

```
F nm,XYZ
```

Console 15 is signed on to the region as *xxxx*CN15 (where *xxxx* is the system user prefix). The XYZ command is then executed in this signed-on environment. Because XYZ is not a command recognized by the region, an NCL procedure of the same name is loaded and executed. If the following statements are in XYZ, then the commands are sent to z/OS as if they were entered one after the other at Console 15:

```
SYSCMD CON=&ZCONSOLE DATA=D J,L
SYSCMD CON=&ZCONSOLE DATA=D U,ALLOC
```

The result of the above commands is then returned to Console 15 without being intercepted by SYSCMD. This is true even when the specified console number is the same as a currently acquired console; the output is sent to the current owner of the console.

Display the Status of the Console Pool

The SHOW CONSOLES command can be used to show the status of the console pool and various statistics for the console pool. The following display provides two types of console information:

- SYSPARMS settings
- Statistics

```
((16.26)------------ Operator Control Services (PROD) -------------------------

show consoles
N86E01 CONSOLE INFORMATION FOLLOWS...
N86E10 SYSPARM SETTINGS...
N86E11 CONSOLES AOMCTYPE AOMCMIGI AOMCUTOK AOMCTO1  AOMCTO2  AOMCTO3
N86E12 (20,10)  EXTMCS   Y        Y        10       20       5
N86E13 AOMCMIGX AOMCOPTS AOMJESCH AOMSUBCH
N86E14 NO       00       $        NO
N86E20 STATISTICS...
N86E21 #-LOG-AQ #-PHY-AQ %-PL-A AV-TM-PA #-LOG-RL #-PHY-RL %-PL-R AV-TM-PR
N86E22       11        7  63.63     1.28        9        0   0.00     0.00
N86E23 #-LA-MIG %-LA-M #-PA-MIG %-PA-M #-LA-FNC %-LA-F #-PA-FNC %-PA-F
N86E24        5  45.45        3  42.85        0   0.00        0   0.00
N86E30 C#  ID  NAME     STATUS   LOCK   AUTH     USERID   ENV      W T/O
N86E31   1 224 ZD1ZZ010 IN-USE   AOM    MASTER   DE1NAOMP AOM      - -
N86E31   2   - ZD1ZZ011 POOL     -      MASTER   -        -        - -
N86E31   3   - ZD1ZZ012 POOL     -      MASTER   -        -        - -
N86E31   4   - ZD1ZZ013 POOL     -      MASTER   -        -        - -
N86E31   5 225 ZD1ZZ014 POOL     -      MASTER   -        -        - -
N86E31   6   - ZD1ZZ015 POOL     -      MASTER   -        -        - -
N86E31   7 246 ZD1ZZ017 POOL     -      MASTER   -        -        - -
N11907 *END*
--------------------------------NetMaster ---------------------------------
==>
```

For more information about the statistics displayed, place your cursor on a line of statistics (for example, the line prefixed by message number N86E21, and press F1 (Help). This displays online help for the statistics on that line.

Use SYSCMD in an &INTCMD Environment

You can use the &INTCMD NCL statement to execute any command in your region in a *dependent processing environment*. The results of commands are queued to the dependent environment and can be read.

To read these results, use the &INTREAD NCL statement.

By using the SYSCMD command in this way, you enable any NCL procedure to issue an operating system command and to receive the results. Command authorization, in this case, is based on the authority of the user initiating the NCL procedure.

Clear a Lock in an &INTCMD Environment

In an &INTCMD environment, a console is released when either &INTCLEAR [TYPE=ALL] is issued, or the executing NCL procedure is terminated. Additionally a SYSCMD OPT=REL could be issued in the INTCMD environment to release the console.

Prefix Messages

SYSCMD messages carry several non-text attributes such as *jobname*. The SYSPARMS command can be used to request the prefixing of several of these attributes to SYSCMD messages, when these are displayed on an OCS console. This prefixing occurs just before display and has no effect on the actual delivery of the messages.

The following message attributes can be prefixed:

- Message time, in the format *hh:mm:ss* or *hh:mm*

  To prefix this, use the SYSPARMS AOMPRFTM=YES/HMS/HM/NO command.

- z/OS job name

  To prefix this, use the SYSPARMS AOMPRFJN=YES/NO command.

- z/OS job ID, in the format JOB *nnnnn*, or STC *nnnnn*, or TSU *nnnnn*

  To prefix this, use the SYSPARMS AOMPRFJI=NO/YES command.

- CA SOLVE:Operations Automation message source domain

  To prefix this, use the PROFILE PREFSYS=YES/NO command. This is not SYSCMD facility-specific.

A MSGPROC can see the text as it will be displayed, by using the &ZMPTEXT system variable.

More information:

SYSPARMS Operands (see page 257)

Facilitate User-Generated Messages

The transportation of user-generated messages is facilitated by the &WTO, &WTOR, and &DOM NCL statements. These statements are available to any NCL procedure and provide a way for an NCL procedure to issue associated WTO, WTOR, or DOM functions.

Insertion Points

The &WTO, &WTOR, and &DOM NCL statements actually issue the associated SVC calls and therefore insert the message (or DOM) into the operating system itself.

&WTO NCL Statement

The &WTO NCL statement provides a direct interface to the WTO macro or SVC.

When &WTO is executed, the &ZDOMID system variable is set to the allocated DOMID. You can use this value on a subsequent &DOM statement to delete the message from the consoles.

Options on &WTO allow settings of the routing code, descriptor code, some MCS FLAG values, and delivery of the message to a specific console.

Recommendations for Using &WTO

When using &WTO, consider the following:

- Always provide a message identifier at the start of the message. This identifier should establish some connection with the issuer of the &WTO.
- Avoid using descriptor codes 1, 2, or 11. These descriptor codes cause the messages to be treated as Non-Roll-Delete (NRD) and can lead to excessive numbers of NRD messages being displayed.
- Excessive use of &WTO can lead to console buffer shortages.

Suggested Uses of &WTO

Suggested uses of &WTO are:

- Making the message Non-Roll-Delete when a major problem is detected, for example:

```
&WTO DESC=1 ROUTCDE=(1,2) DATA=C999 MAJOR CATASTROPHE +
                                    OCCURRED
&SAVEDOMID = &ZDOMID -* save domid for later deletion
```

  This message remains on all consoles until deleted by a following &DOM.

- Issuing a message to the master console, for example:

```
&WTO CONSOLE=1 DATA=AOM001 HI THERE...
```

- Issuing a console broadcast, for example:

```
&WTO MCSFLAG=BC DATA=BC001 Broadcast msg
```

&WTOR NCL Statement

The &WTOR NCL statement provides a direct interface to the WTOR macro or SVC.

When &WTOR is executed, the NCL procedure is suspended until a reply to the WTOR is received or until an optional WAIT time (in seconds) has expired (the WTOR is canceled in this case). If the NCL procedure is flushed, the WTOR is also canceled.

Options on &WTOR allow the setting of routing codes, some MCS FLAG values, and delivery of the message to a specific console.

Other options allow the setting of a maximum reply length and an indication of how the reply is to be returned. It can be broken into words, nominated variables, or as a string in &1.

Recommendations for Using &WTOR

When using &WTOR, consider the following:

- Always provide a message identifier at the start of the message. This identifier should establish some connection with the issuer of the &WTOR.
- WTOR messages are always treated as Non-Roll-Delete. For this reason, keep the number of WTOR messages outstanding at any one time to a minimum.

Suggested Use of &WTOR

Use &WTOR to enable any NCL procedure to carry on a dialog with the z/OS console operator or with outboard automation tools such as CA Automation Point:

```
&WTOR STRING DATA=MSG001 WHAT IS YOUR NAME?
&WTO MSG002 HI, &1, I AM AOM
```

&DOM NCL Statement

The &DOM NCL statement provides a direct interface to the DOM macro or SVC.

The NCL procedure must supply a valid DOMID to the &DOM statement. This DOMID can be obtained from a preceding &WTO (using the &ZDOMID system variable).

A DOMID is formatted as eight hexadecimal digits. The first two are the system ID and the last six are a z/OS-assigned message ID.

When the DOM is received, it deletes any WTO or WTOR with the matching DOMID from the consoles.

Recommendations for Using &DOM

When using &DOM, consider the following:

- Issue &DOM only with a valid DOMID obtained as described above. An invalid format DOMID (not eight hex digits) causes the NCL procedure to be terminated. If the DOMID appears valid, but the actual number is not valid, the wrong message may be deleted.
- Delete messages only for a good reason. Indiscriminate deletion of critical messages can lead to severe operational problems.

Suggested Use of &DOM

Use &DOM following an &WTO statement to delete a message sent with descriptor code 1, 2, or 11. The &DOM is sent when the condition that caused the original message to appear is resolved.

Example

```
&WTO ...
&SAVEDOMID = &ZDOMID
...
...
&DOM ID=&SAVEDOMID
```

SYSCMD Facility Commands

The SYSCMD facility contains the following commands:

AOM START

Starts the local SYSCMD subset of the AOM operating system interface.

AOM STOP

Stops the local SYSCMD subset of the AOM operating system interface.

PROFILE

Displays a user profile.

SHOW AOMABEND

Displays diagnostic information if the SYSCMD subsystem interface code abends.

SHOW AOMSTAT

Displays AOM statistics.

SHOW CONSOLES

Displays the consoles currently allocated to your system for use by the SYSCMD facility.

STATUS

Displays current system status information.

SUBMIT

Passes a command to a background environment for processing.

SYSCMD

Issues a command to the local operating system and returns associated response messages to your system.

SYSPARMS

Initializes or modifies SYSCMD facility system parameter values.

Note: For more information about the above SYSCMD facility commands, see the online help.

SYSCMD NCL Verb and System Variable Summary

The following NCL verbs and system variables are available:

&DOM

Generates a z/OS DOM (Delete-Operator-Message).

&WTO

Generates a z/OS or WTO (Write-To-Operator) message.

&WTOR

Generates a z/OS WTOR (Write-To-Operator with Reply) message and waits for a reply.

The following system variables are available:

&ZMAOMAU

Inserts an AOM message authorized issuer flag.

&ZMAOMBC

Generates an AOM message broadcast flag.

&ZMAOMDTA

Indicates whether AOM data is present.

&ZMAOMID

Inserts AOM ID.

&ZMAOMJI

Inserts AOM job ID for z/OS-sourced messages.

&ZMAOMJN

Inserts AOM job name for z/OS-sourced messages.

&ZMAOMMID

Inserts AOM message ID.

&ZMAOMMIN

Inserts Multi-Line Write To Operator (MLWTO) minor line flag.

&ZMAOMMLC

Indicates YES if MLWTO minor CTL line.

&ZMAOMMLD

Indicates YES if MLWTO DATA.

&ZMAOMMLE

Indicates YES if MLWTO END.

&ZMAOMMLL

Indicates YES if MLWTO LABEL.

&ZMAOMMLT

Has MLWTO minor line type.

&ZMAOMMLV

Indicates AOM message level.

&ZMAOMMSG

Indicates whether message was delivered to an AOM receiver.

&ZMAOMRC

Inserts AOM routing codes in list format.

&ZMAOMRCM

Inserts AOM routing codes in MASKCHK format.

&ZMAOMRCX

Inserts AOM routing codes in hexadecimal format.

&ZMAOMSOS

Inserts AOM message source operating system environment.

&ZMAOMTM

Inserts AOM message time.

&ZMAOMTYP

Inserts AOM message type.

&ZMAOMUFM

Inserts AOM user flags in MASKCHK format.

&ZDOMID

Inserts CA SOLVE:Operations Automation or z/OS DOMID.

&ZFDBK

Inserts feedback code set by several NCL verbs.

&RETCODE

Inserts return code set by several NCL verbs.

Note: For more information about these SYSCMD NCL verbs and system variables, see the *Network Control Language Reference Guide*.

Chapter 6: Administering Signed-on Users

This section contains the following topics:

Show Active Users (see page 79)

Show Active Users

At any time, only a selection of the defined users are actively signed-on to, or disconnected from, a given system. You can display these user IDs using *one* of the following methods:

- SHOW USERS command in the OCS window
- Active User List Facility

Note: All users must be defined to the security system by using *one* of the following:

- The User ID Access Maintenance Subsystem (UAMS)
- An external security package

For more information about security, see the *Security Guide*.

SHOW USERS Command

To show active users

1. From OCS, enter **SHOW USERS** at the prompt.

   Note: For more information about this command, press F1 (Help).

Active User List Facility

The Active User List facility assists help desk staff or the systems administrator to perform the following functions for one or more domains connected by INMC links:

- Monitor user activity
- Cancel a user
- Disconnect a user

The System Support : User ID List lets you identify which domain (in the generic resource) each user is currently attached to. You can then apply any of the available actions to a particular user attached to a particular domain.

On the User ID List, entries are displayed within their domains, with delimiter lines dividing the domains. The local domain (if selected) is always shown first, with others following in domain name (link name) order.

Error messages, such as NO MATCHING USER(S) ON THIS DOMAIN, are displayed for any domain where they apply.

Matching user IDs show the user ID and one of the following:

- The terminal name (LU name) if logged on (signed-on user)
- The disconnection data if applicable (disconnected user)

To list active users

1. Enter **/SS** at the prompt.

   The Security and System Services : Primary Menu appears.

2. Type **LU** at the Select Option prompt and complete the following fields:

   User ID

   Specifies the user ID for which you want to search. You can enter the leading characters of a user ID to limit the search. If you enter eight characters, then this value is used as an exact match. If you enter less than eight characters, then this value is treated as a prefix.

   Note: If the last character is an *, then it is ignored; that is, user IDs USER01 and USER01* are equivalent.

   Link or Domain Name

   Identifies the name of the domain from which you want to obtain information. There are four options:

   - Leave blank for the local system.
   - Enter a specific link name.
   - Enter **?** to display a list of link names from which you can select one or more.
   - Enter ***** to denote all active remote domains.

   Note: Link name on this panel means an INMC link that is currently active.

   Press Enter.

   The System Support : User ID List appears. The panel provides actions that enable you to disconnect or cancel a user session.

Chapter 7: Customizing External Applications Access

This section contains the following topics:

External Applications Function (see page 81)

External Applications Function

The external applications function lets you access other products to assist in monitoring resources. It lets you specify access to external applications for network management, configuration management, problem management, and help desk.

Access External Applications Function

To enable access to external applications from the region

1. Enter **/PARMS** at the prompt.

   The Customizer : Parameter Groups panel appears.

2. Enter **U** (Update) beside the EXTAPPLS parameter group in the Interfaces category.

   The Customizer : Parameter Group panel for EXTAPPLS appears.

3. Complete the following fields for each type of external application that you want to access:

   LU1 Logmode

   Specifies the logon mode to be used by MTO command sessions.

   Note: These are the terminal name prefixes for the VTAM APPLS as defined during your installation.

4. Press F8 (Forward) to scroll through the next two panels.

   The application definition panel appears.

5. Complete the following fields for each external application that you want to access:

   Application Name

   Specifies the network name of the application providing the associated function.

   Logon Script

   Specifies the name of a procedure used to automate the application logon sequence and also request information from the application about a resource.

   Logon Data

   Specifies the data supplied to the external application as a part of session startup.

   a. (Optional) If you want to apply the changes immediately, press F6 (Action).

      The region takes on the changes.

   b. Press F3 (File).

      The changes are saved.

Access Applications

When the applications are specified, you can access them using the following commands:

CFG

Accesses the configuration management application known to the region.

HD

Accesses the help desk application known to the region.

PRB

Accesses the problem management application known to the region.

XNM

Accesses the external network management application known to the region.

Navigate to an External Application

To customize where you log on to your external application, you can use a logon script. A logon script lets you automate the logon process.

Logon scripts are supplied for the following products:

RMSCPTNM

CA Mainframe Network Management

RMSCPTPM

CA SOLVE:Central Problem Management

RMSCPTCM

CA SOLVE:Central Configuration Management

RMSCPTHD

CA SOLVE:Central Problem Management Help Desk

Customize Logon Scripts

You can customize logon scripts to meet your own application requirements.

To customize a logon script

1. Enter **/ASADMIN.S** at the prompt.

   The Logon Scripts panel appears.

2. Enter the letter of the required Generate option at the Select Option prompt.

   The Logon Recording : Recording Details panel appears.

3. Specify the output string. This is the string that appears on the primary menu of the application that you are logging on to.

4. Specify the resource name string. This string will be recognized during recording as the point to substitute a resource name for which the command is to be executed.

5. Specify the stop key. The default is F15. This key ends the logon recording session.

6. Press F6 (Action).

   The logon recording starts.

7. Log on to the application, and do *one* of the following:

   - Proceed to the required location and press the stop key.
   - Proceed to a location where a resource name is required and enter the resource name string as specified in Step 4. Continue further if desired, and then press the stop key.

   The Logon Recording : Save Script panel appears.

8. Enter the data set details and press F6 (Generate).

   The script is generated and displayed.

Chapter 8: Customizing and Using MAI-OC

This section contains the following topics:

MAI-OC (see page 86)
Define MAI-OC to Target Applications (see page 87)
Cross-Domain MAI-OC Sessions (see page 90)
Log On to Another Application (see page 90)
Log Off an Application (see page 93)
Disconnect an MAI-OC Session (see page 93)
Interrupt an MAI-OC Session (see page 93)
Send Data to an Application (see page 94)
Receive Data from an Application (see page 94)
Issue Commands to an Application (see page 95)
Use MAI-OC with Multiple Regions (see page 96)
Set EQUATE Values for MAI-OC Commands (see page 97)
Use an MAI Installation Exit (see page 98)
Session Protocols (see page 98)
SCS Character Support (see page 99)
Strike-over Masks (see page 101)
JES MAI-OC Sessions (see page 102)
MAI-OC Mode Table and Bind Checks (see page 102)
MAI-OC Operational Scenario (see page 104)

MAI-OC

MAI-OC lets you start multiple sessions with VTAM applications using Logical Unit Type-1 (LU1) protocols. It is available from OCS or from an NCL procedure. MAI-OC appears to the application as a line-by-line device, such as an IBM 3767 terminal.

You can use MAI-OC to provide centralized network operation of major systems such as CICS, IMS, or JES. The MAI-OC sessions act as the master consoles of the other application systems.

MAI-OC facilities are available from any Operator Console Services (OCS) window; however, before using MAI-OC, you should consider the use of MAI-OC sessions with certain subsystems, cross-domain MAI-OC sessions, and the use of MAI-OC from multiple systems to the same target applications.

You can operate an MAI-OC session from an NCL procedure using standard internal command environment processing through the &INTCMD facility. The NCL procedure can send data across the MAI-OC sessions that it is maintaining and receive output from those sessions.

MAI-OC sessions with other applications can be created from any processing environment. Most things that can be done from a native terminal can be done using an MAI-OC session.

In this section there are references to the commands that control the operation of MAI-OC. The use and syntax of MAI commands are described in the online help.

Define MAI-OC to Target Applications

When an MAI-OC session is established with a target application, MAI-OC emulates an LU-Type 1 device (for example, an IBM 3767) as the secondary end of the session. The target application sees the MAI-OC connection as a standard session with a physical 3767 terminal.

Certain application subsystems, such as IMS, require that every logical unit with which they are to have a session be defined to them before any session with the LU is allowed.

To establish MAI-OC sessions with systems such as CICS or IMS, include the VTAM LU names for MAI-OC to use when requesting the session in the appropriate system definitions. Other relevant information should be included with the VTAM LU names, such as the ability of the logical unit to act as a master terminal, its authority level.

The definition to a subsystem such as CICS or IMS is the same as for a physical 3767 device. See the appropriate manuals for the precise coding requirements for the system that you are using.

Sample LU Definitions

The following sections provide sample LU definitions for CICS, IMS, JES2, and JES3.

Note: All LU names likely to be used to create a session with these applications should be defined as a separate terminal to that application.

For information about definition requirements and to customize the definitions to your own requirements, see the appropriate subsystem guides.

Define MAI-OC to a CICS System

The following definition enables an MAI-OC session to start with a CICS system using an LU name of NMMAO001:

```
DFHTCT   TYPE=TERMINAL                                    *
         ACCMETH=VTAM                                     *
         BRACKET=YES                                      *
         BUFFER=256                                       *
         BMSFEAT=(noroute,norouteall)                     *
         GMMSG=YES                                        *
         NETNAME=NMMAO001                                 *
         OPERID=id                                        *
         OPERPRI=code                                     *
         PGESIZE=(12,80)                                  *
         PGESTAT=PAGE,*RELREQ=(YES,YES)                   *
         RUSIZE=256                                       *
         TIOAL=256                                        *
         TRMIDNT=term                                     *
         TRMSTAT=TRANSCEIVE                               *
         TRMTYPE=3767
```

Define MAI-OC to an IMS System

The following definition enables an MAI-OC session to start with an IMS system using the LU name of NMMTO and providing master terminal authority:

```
TYPE      UNITYPE=SLUTYPE1
TERMINAL  NAME=NMMTO,COMPT1=(CONSOLE,MFS-SCS1),OUTBUF=256
NAME      (lterm,MASTER)
```

The following definition provides IMS support for a terminal named NMMAO001, which could be used by MAI-OC for general operations and transaction execution:

```
TYPE      UNITYPE=SLUTYPE1
TERMINAL  NAME=NMMAO001,COMPT1=(CONSOLE,MFS-SCS1),OUTBUF=25
          6
NAME      lterm
```

Define MAI-OC to a JES2 System

The following definition lets you use MAI-OC on a JES2 system:

```
LOGON1 APPLID=JES2
&MAXSESS=nn
&NUMLNES=nn
&NUMRJE=nn
&NUMTPBF=nn
&MLBFSIZ=256
&TPBFSIZ=256

LINE1       UNIT=SNA

RMT1        LUTYPE1,BUFSIZE=256,LINE=1,CONSOLE,                    *
            NOCMPCT,NOCOMP,SETUPHDR,                               *
            SETUPINF,WAITIME=1,                                    *
            NUMPR=0,NUMRD=0,LUNAME=NMMAO001
```

In this example, an MAI-OC session started on LU NMMAO001 is automatically assigned to workstation RMT1. If the LUNAME parameter on the RMT1 statement is omitted, a user creating an MAI-OC session can specify the required workstation name in user data on the MAILOGON command, for example:

```
MAILOGON JES2 DATA=RMT1
```

Define MAI-OC to a JES3 System

The following definition lets you use MAI-OC on a JES3 system:

```
COMMDEFN,APPLID=JES3,LU=nn
CONSOLE,JNAME=RJE01,TYPE=RJP,DEST=NONE
RJPWS,N=RJE01,RD=0,PR=0,G=RJE01,AUTO=N,
COMPACT=NO,C=R,LU=NMMAO001
```

In this example, an MAI-OC session started on LU NMMAO001 is automatically assigned to workstation RJE01. If the LU parameter on the RJPWS statement is omitted, a user creating an MAI-OC session can specify the required workstation name in user data on the MAILOGON command, for example:

```
MAILOGON JES3 DATA=RJE01
```

Cross-Domain MAI-OC Sessions

Unless VTAM is configured to support dynamic cross domain definition and adjacent SSCP lookup, for a user in one domain to request an MAI-OC session with a target application running in another domain, the following conditions must be satisfied:

- The appropriate cross domain resource definitions must be filed in the VTAM definition library
- The target subsystem must have the MAI-OC LUNAME defined to it as a valid terminal if required

Log On to Another Application

To log on to another application using MAI-OC, use the MAILOGON command. Logging on creates an MAI-OC session. Sessions can be created with as many applications as you require and multiple sessions can be created with the same application.

For example, to establish a session with a CICS application with the application ID of CICSA, enter the following command:

```
MAILOGON CICSA
```

When the connection is established, you receive an initial message from the application informing you that you are connected.

Note: If the session is established with JES, you do not receive a message to confirm connection.

Create a Session Identifier

Each session you create is given a unique session identifier. This identifier defaults to the name of the application program with which the session is established, but you can choose any 1- to 8-character name.

For example, to change the session ID of your CICS application from CICSA to CICSPROD, enter the following command:

```
MAILOGON CICSA ID=CICSPROD
```

The session identifier is used in all MAI-OC commands, so you should make it as meaningful as possible.

If you create multiple sessions with one application from the same window, you must allocate a unique identifier to each session. If the first session's identifier defaults to the application name, you must specify a unique identifier for subsequent sessions with the same application.

The uniqueness of a session identifier applies only to the primary environment. You can open another window at the terminal and create more MAI-OC sessions using identifiers already used in the first window.

List Established Sessions

To list all active MAI-OC sessions, enter the SHOW MAI command. This lets you list all of the MAI-OC sessions that are established and the identifiers used for each.

Select an LU Name for an MAI-OC Session

Before you can start an MAI-OC session with a target application, the system must allocate the LU name that is to be used to act as the *terminal* end of the session.

The LU name may be allowed to default or a particular LU name may be specified on the LU= operand of the MAILOGON command.

Choose an LU Name From a Pool

If no specific LU name is provided on the MAILOGON command, MAI-OC generates one consisting of the MAIOPREF prefix (set by the LU1 Terminal Prefix field of the EXTAPPLPOOLS parameter group in Customizer) followed by a number in the range 001 - 999. The number chosen is the first that is not already in use by another MAI-OC session.

This technique lets you establish an MAI-OC session without knowing the identity of the terminal that MAI-OC will simulate. It also implies that when the session is established, the LU name used by MAI-OC is unpredictable.

When an MAILOGON request without a specified LU name fails because MAI has chosen an unknown LU name, MAI tries up to five successive LU names (each beginning with the MAIOPREF prefix) before indicating that no MAI-OC LUs are available. This is because a defined MAI-OC LU may have been varied inactive; therefore, appears to MAI the same as an LU that has not been defined.

Choose a Specific LU Name

If you want an MAI-OC session with a target application in which the MAI-OC LU name must be the name of a specific terminal, the MAILOGON LU= operand lets you specify the LU name that MAI-OC is to use.

This technique requires that you have knowledge of the terminal that is to be used on the session, but it also means that the identity of the terminal is predictable.

This facility is necessary to establish an MAI-OC session that has to have particular attributes, for example:

- An IMS system is generated with its IMS Master Terminal (primary operating console) having the LU name of MTO3767P.
- An operator is to operate an MAI-OC session from the OCS window of a terminal, with the MAI-OC session driving the IMS Master Terminal.
- The Operator requests the MAI-OC session with a MAILOGON command, specifying LU=MTO3767P on the command text. MAI-OC will open a VTAM ACB whose APPL name is MTO3767P and which must have been defined to VTAM as an APPL.

Note: An MAI Installation Exit (MAIEX02), if provided, is driven whenever a session request is processed. This exit may override the LU name or prefix, if required.

Log Off an Application

Most MAI-OC sessions can be terminated by sending a logoff command of the type expected by the application. For example, for TSO this would be LOGOFF, for IMS it would be /RCL.

However, some applications (such as JES) do not have a logoff command, or you may have trouble sending the command. In these cases, you can use the MAIDISC command to force the disconnection (see page 93).

If you exit OCS with MAI-OC sessions intact, your region automatically generates MAIDISC commands for all your MAI-OC sessions. This causes lost terminal conditions at the applications for all your MAI-OC sessions. We do not recommend that you use MAIDISC to end TSO sessions because the logoff leaves a reconnect environment pending for a system-defined period.

Disconnect an MAI-OC Session

If you cannot log off an MAI-OC session normally, you can use the MAIDISC command and specify the session ID that you want to disconnect with. For example, to end the session with the CICSA application that was established above, enter the following command:

```
MAIDISC CICSA
```

Interrupt an MAI-OC Session

You can interrupt an MAI-OC session to achieve different effects depending on the application you are connected to. For example, if you have a TSO session established, an interrupt cancels the current operation. If you have an IMS session, it removes the current message from the queue and requests the next one. See the relevant product guide for more information about the effect of an interrupt.

To interrupt an MAI-OC session, use the MAIINT command. For example, to interrupt the session established with CICSA, enter the following command:

```
MAIINT CICSA
```

The MAIINT command can also be specified to generate an attention interrupt to the application by using the TYPE=ATTN operand, and a cancel interrupt by using the TYPE=CNCL operand.

Send Data to an Application

When you have logged on to an application, you may want to send data to the application.

To send data to an application, use the MAISEND command. This command nominates the session identifier of the session over which you want to send the data, followed by the data you want to send. For example, to send data to the CICSA session, enter the following command:

```
MAISEND CICSA CEMT I TRAN
```

MAI-OC may append a new line character to the message (to simulate a RETURN key) and the data is sent.

Note: An MAI-OC session looks like a session with a hard-copy terminal. It does not function on a full screen basis. For example, an attempt to invoke ISPF on a TSO session is rejected.

Receive Data from an Application

Data received from an application is issued as line messages to the environment that last issued an MAI-OC command against that session. For example, if you issued an MAILOGON command from OCS, then the initial application messages resulting from the session establishment are received by the OCS environment. If the next MAI-OC command for that session is from a dependent environment, for example MAISEND through $CMDENT, then further messages are returned to the dependent environment.

Application data is displayed unchanged, with the possible addition of some information before the text. This information is the session identifier of the session from which the data was received. For example:

```
(CICS) H2002I TERMINAL CONNECTED
```

The presence of the prefix information and its format is controlled by options specified on the MAILOGON command.

The messages may be in response to a command or NCL system variable that was issued, or they might be unsolicited information, depending on the way the application functions. However, all messages are flagged as unsolicited.

Note: The PROFILE UNSOL=NO command does not prevent the receipt of messages generated by an MAI-OC session.

Issue Commands to an Application

When you issue commands to an application using an MAI-OC session, it works in the same way as issuing commands using OCS. However, there are some special considerations for sessions with IMS applications.

MAI-OC lets you simulate a logical keyboard locked condition, as well as abbreviate commands, use function keys, and use NCL procedures to simplify control procedures for the application.

Any commands that can be issued from an OCS window can also be issued from an NCL procedure. Even NCL procedures operating in full-screen mode (for example, invoked through an FSPROC command) can make use of MAI-OC sessions.

Issue Commands While Waiting for Application Response

Because MAI-OC simulates a real terminal, it is possible to get a logical keyboard locked condition in which MAI-OC is, for instance, waiting for a response from the application. At this time, the MAISEND command cannot be used to send data to the application and if entered is rejected with an appropriate error message. Normally the command can be retried later. Of course, any other region commands can be entered while you are waiting.

Abbreviate Commands

NCL procedures and terminal function keys can be used to simplify MAI-OC command requirements, and many MAI-OC commands can be shortened or automated using the EQUATE command.

It is possible that EQUATE commands and NCL procedures have already been set up that you can use. Check with your systems administrator.

Issue Multi-Segment Commands to an IMS Application

When sending commands to an IMS application over an MAI-OC session, IMS requires that some input messages be multi-segment. Specifically, a /BRO command must be in two segments. Consider, for example, the following command:

```
/BRO NODE NMMAV003 COFFEE TIME
```

To make this command form two segments, IMS requires a new line character after the node name and before the message.

The MAILOGON command lets you specify a character to represent a new line character in data sent through an MAISEND command.

For example, you can create an MAI-OC session using the following command:

```
MAILOGON IMS NL=+
```

This command defines the new line character as the plus sign (+). You can now send the above broadcast command by entering the following command:

```
MAISEND IMS /BRO NODE NMMAV003+COFFEE TIME
```

The plus sign (+) is replaced by the necessary new line character.

Use MAI-OC with Multiple Regions

When more than one region in a network has MAI-OC sessions with the same set of target applications, you should assign a unique set of MAI-OC LU names to each region. This associates the names used by each region with the region in which that region is executing, and avoids VTAM definition conflicts when attempting to start cross-region sessions.

The other advantage of assigning each region its own set of MAI-OC LU names is that it allows added control over which regions can establish MAI-OC sessions with which subsystems.

Set EQUATE Values for MAI-OC Commands

The MAI-OC feature performs the functions of session connection and disconnection and sends messages on sessions in response to MAI-OC commands issued by users.

MAI-OC commands can be made easier to use if the standard SOLVE EQUATE command is utilized. For certain applications, use of EQUATE values enables the operation of MAI-OC sessions to be identical to operation of the same session from a native terminal attached directly to the application.

For example, if you have an MAI-OC session with an IMS system and want to display the IMS transaction queue, you would use the following full MAI command:

```
MAISEND IMS /DIS Q TRAN
```

MAISEND

Specifies the command that requests MAI-OC to transfer data across a session.

IMS

Specifies the session identifier of a session with the IMS application to which the message is to be sent.

The remainder of the data, starting with the slash (/), is the message to be sent to the application.

By setting up an EQUATE value as:

```
EQUATE / MAISEND IMS /
```

you can enter the IMS message text in its native form and allow the system to expand the slash (/) to the full MAISEND command format.

An alternative may be to EQUATE the target application name to the MAISEND command.

Another example, for use on a JES session, could be:

```
EQUATE $ MAISEND JES2 $
```

You can use it to enter JES2 commands as if in native mode, for example:

```
$DA
```

Note: The use of EQUATEs varies according to the requirements of different installations, and you should consider how EQUATEs can be used to make MAI-OC operation simple in your installation.

Use an MAI Installation Exit

An MAI installation exit (MAIEX02) is provided with your product. This provides security checking and validates and changes the characteristics of an MAI-OC session.

The exit is driven whenever a session request is to be processed if the MAIEX02 SYSPARM has an exit name specified. This exit can override the LU name or prefix, if required.

More information:

MAI Installation Exit MAIEX02 (see page 295)

Session Protocols

An MAI-OC session functions as a true SNA LU-type 1, and adheres to the protocols described in the IBM publication *3767 Component Description*.

When MAI-OC has a session with JES2 or JES3, it appears as a 3776-type RJE device. This is still an LU-type 1, and the protocols used are a subset of those described above.

As MAI-OC is simulating a real terminal, it is possible to get a logical *keyboard locked* condition in which MAI-OC is, for instance, waiting for a response from the application. At this time, the MAISEND command cannot be used to send data to the application, and if entered will be rejected with an appropriate error message.

The SHOW MAI command can be used to determine the session states of MAI sessions. Information given includes whether the *keyboard* is locked or unlocked, the SNA bracket state, and the general session state. Abbreviations used for states in the display are generally those used in the *3767 Component Description*:

INB

In Bracket

BETB

Between Brackets

BBP

Begin Bracket Pending

SEND

Send State (can send to application)

RCV

Receive State (cannot send)

DRWT

Waiting for a definite response (cannot send)

STBY

Standby State (can send).
Indeterminate state (state change in progress or not in session).

The CON field in the display may contain the following:

YES

Session established and available.

NO

Session not yet established.

LCK

Session established but keyboard locked, because session state is not such that data may be sent.

SCS Character Support

SCS control characters are used by some systems for print layout instructions. They tell an output device (usually a printer) how to respond to tab, spacing, line break and other formatting control sequences.

Not all SCS control characters can be fully simulated at a terminal; however, none cause a session to be rejected, and wherever possible MAI-OC translates the SCS character to the best equivalent function that OCS mode can provide.

This section details the actions taken by MAI-OC on receiving data streams containing the following SCS characters:

New Line X'15'

Data following the character is displayed on a new line of the operator window.

Form Feed X'0C'

As New Line.

Line Feed X'25'

Stripped from the data stream.

Vertical Tab X'0B'

As New Line.

Record Separator X'1E'

As New Line.

Carriage Return X'0D'

As New Line.

Vertical Channel Select X'04nn'

As New Line.

Horizontal Tab X'05'

Replaced by a blank.

Backspace X'16'

Logically deletes previous character in the line.

Inhibit Print X'24'

Stops data sent to the application being echoed to the screen or the activity log. Data is replaced by asterisks.

Enable Print X'14'

Resumes echoing after a previous Inhibit Print.

Set Horizontal Format X'2BC1'

Stripped from the data stream (together with all associated counts and so on).

Set Vertical Format X'2BC2'

As Set Horizontal Format.

SCS Characters Sent by MAI-OC

The only SCS character sent by MAI-OC to an application is the New Line (X'15') character. It is appended to each message sent and you can be embedded in data. For more information, see the NL= operand of the MAILOGON command in the online help.

Strike-over Masks

A common technique used on hard copy terminals to hide entered data such as passwords is the use of a strike-over mask, where two or more lines of characters are printed one over the other, and the print head left underneath these characters. The next line of data typed is then unreadable.

MAI-OC keeps track of where the print head would be on a real hard copy terminal and prevents the echoing to the screen or activity log of all or some of the next line of data sent to the application. Any characters sent that would be underneath other characters are replaced by an asterisk. For example, suppose MAI-OC received the following string of characters from the application:

XXXXXXXX<LLLLLLLL<00000000<

<

Specifies an SCS carriage return (X'0D').

If the MAI-OC user were then to send the characters MYPASSWORD to the application, those characters would be echoed to the screen and log as ********RD, because the first eight characters would be obscured on a real terminal. Multiple backspace characters instead of carriage return could be used in the mask.

An alternative to the use of strike-over masks is the use of the Inhibit and Enable Print SCS control characters.

JES MAI-OC Sessions

JES regards an MAI-OC session as a session with an RJE workstation. This means that data sent to JES is regarded as input from a remote console, so any authorized JES command may be sent and the results returned to the window.

However, JES does limit the scope of commands that can be entered from a remote console. Generally, with the provision of the appropriate operands on commands and the correct authorization in JES, commands can be entered to perform any JES display-type function. However, commands can only change the status of jobs, and so on, owned by the workstation. Of course, the OPSYS OCS command can be used to enter commands, if required.

JES commands are available to shorten responses to commands, for example, to remove the leading time stamps). Their use should be considered to make the display as neat as possible.

JES2, and JES3 if so configured, do not send a salutation message to a workstation when it logs on. This means that there is no indication that an MAI-OC session request has completed. Use the SHOW MAI command to determine when the session is established.

JES does not have a logoff command. Use the MAIDISC command to terminate a JES session.

MAI-OC Mode Table and Bind Checks

The following logmode table should be assembled and linked into the appropriate VTAM library (for example, SYS1.VTAMLIB in z/OS). It accurately defines MAI-OC session characteristics and results in the most efficient use of a session. It should then be specified on all MAI-OC VTAM APPL statements using the MODETAB=MAIVMODE operand:

```
MAIVMODE  MODETAB

MAIVMDE   MODEENT LOGMODE=MAIVMDE                                    *
          FMPROF=X'03'                                               *
          TSPROF=X'03'                                               *
          PRIPROT=X'B1'                                              *
          SECPROT=X'90'                                              *
          COMPROT=X'3080'                                            *
          RUSIZES=X'8585'                                            *
          SSNDPAC=X'00'                                              *
          SRCVPAC=X'01'                                              *
          PSNDPAC=X'01'                                              *
          PSERVIC=X'010000008000800000000000'
```

The following table shows the checks MAI-OC performs on BIND parameters at session initiation. The bits shown are checked by MAI; bits not shown are not checked. Invalid BIND parameters are rejected by MAI.

Byte	Bit	Setting	Meaning
2	all	X'03'	FMPROF
3	all	X'03'	TSPROF
4			PRIPROT
	2-3	B'00'	Invalid
		B'01'	Exception response
		B'10'	Definite response
		B'11'	Exception or definite response
	6	B'0'	Compression not used
	7	B'1'	End bracket may be sent
5			SECPROT
	2-3	B'00'	Invalid
		B'01'	Exception response
		B'10'	Definite response
		B'11'	Exception or definite response
	7	B'0'	End bracket not sent
		B'1'	End bracket may be sent
6			COMPROT1
		2B'1'	Brackets are used
	3	B'1'	Bracket termination rule 1
	4	B'0'	Alternate code not used
7			COMPROT 2
	0-1	B'00'	Invalid
		B'10'	Flip-flop mode
		B'01'	Contention mode
		B'11'	Invalid
	2	B'0'	Contention loser recovers
	3	B'0'	Primary is contention loser

MAI-OC Logmode Entry Selection

MAI chooses the logmode entry for an MAI-OC session by searching (by name) the logmode table specified by the MODETAB operand in the APPL definition for the MAI-OC LU selected. The logmode table specified by this operand must be assembled and linked into a load library accessible to CA SOLVE:Access.

The MAIVMODE table supplied in the CC2DSAMP distribution library contains a sample logmode entry used by MAI for MAI-OC sessions. It is recommended that this entry be copied into the logmode table specified. Alternatively, the MAI-OC APPL definition may specify the MAIVMODE table. This is the case in the sample MAI-OC APPL definitions.

MAI-OC Operational Scenario

This section contains examples of VTAM and MAI-OC definitions that are necessary in an installation with the following configuration and requirements:

- Two regions, one called NMP running on the production machine, the other called NMT running on the testing machine
- TSO running on both machines, one called TSOP, the other TSOT
- IMS running on both machines, one called IMSP, the other IMST
- A Network Operator uses a terminal logged on to NMP to:
  - Control VTAM in both machines
  - Operate IMS Master Terminals to both IMS systems using MAI-OC from his terminal
- Authorized personnel can log on to either region and create MAI-OC sessions with the TSO or IMS of their choice. A maximum of three MAI-OC sessions from each region are allowed.

Production Machine Definitions

This section contains examples of the following production machine definitions:

- VTAM
- MAI-OC
- IMSP

VTAM Definitions

```
MAOP001 APPL    MODETAB=MAIVMODE,EAS=1
MAOP002 APPL    MODETAB=MAIVMODE,EAS=1
MAOP003 APPL    MODETAB=MAIVMODE,EAS=1
MAOMTOP APPL    MODETAB=MAIVMODE,EAS=1
MAOMTOT APPL    MODETAB=MAIVMODE,EAS=1

MAOT001 CDRSC CDRM=TCDRM
MAOT002 CDRSC CDRM=TCDRM
MAOT003 CDRSC CDRM=TCDRM
```

MAI-OC Definitions

The value of the LU1 Terminal Prefix field in the EXTAPPLPOOLS parameter group is NMMAV.

IMSP Definitions

```
TYPE        UNITYPE=SLUTYPE1
TERMINAL    NAME=MAOMTOP,COMPT1=(CONSOLE,MFS-SCS1),
            OUTBUF=256
NAME        (MAOMTOP,MASTER)
TERMINAL    NAME=MAOP001,COMPT1=(CONSOLE,MFS-SCS1),
            OUTBUF=256
NAME        MAOP001
TERMINAL    NAME=MAOP002,COMPT1=(CONSOLE,MFS-SCS1),
            OUTBUF=256
NAME        MAOP002
TERMINAL    NAME=MAOP003,COMPT1=(CONSOLE,MFS-SCS1),
            OUTBUF=256
NAME        MAOP003R
TERMINAL    NAME=MAOT001,COMPT1=(CONSOLE,MFS-SCS1),
            OUTBUF=256
NAME        MAOT001
TERMINAL    NAME=MAOT002,COMPT1=(CONSOLE,MFS-SCS1),
            OUTBUF=256
NAME        MAOT002
TERMINAL    NAME=MAOT003,COMPT1=(CONSOLE,MFS-SCS1),
            OUTBUF=256
NAME        MAOT003
```

Network Operator Action

The following creates an MAI-OC session to IMSP as Master Terminal:

```
MAILOGON IMSP LU=MAOMTOP NL=+ WAIT=PERM
```

The following creates a cross-domain MAI-OC session to IMST as Master Terminal:

```
MAILOGON IMST LU=MAOMTOT NL=+ WAIT=PERM
```

Other User

The following creates a session with IMSP using LU MAOP001:

```
MAILOGON IMSP
```

The following creates a cross-domain session with IMST using LU MAOP002:

```
MAILOGON IMST
```

The following creates a cross-domain session with TSOT using LU MAOP003:

```
MAILOGON TSOT
```

Testing Machine Definitions

This section contains examples of the following testing machine definitions:

- VTAM
- MAI-OC
- IMST

VTAM Definitions

```
MAOT001 APPL    MODETAB=MAIVMODE,EAS=1
MAOT002 APPL    MODETAB=MAIVMODE,EAS=1
MAOT003 APPL    MODETAB=MAIVMODE,EAS=1

MAOP001 CDRSC CDRM=PCDRM
MAOP002 CDRSC CDRM=PCDRM
MAOP003 CDRSC CDRM=PCDRM
MAOMTOT CDRSC CDRM=PCDRM
```

MAI-OC Definitions

The value of the LU1 Terminal Prefix field in the EXTAPPLPOOLS parameter group is MAOT.

IMST Definitions

TYPE	UNITYPE=SLUTYPE1
TERMINAL	NAME=MAOMTOT,COMPT1=(CONSOLE,MFS-SCS1), OUTBUF=256
NAME	(MAOMTOT,MASTER)
TERMINAL	NAME=MAOP001,COMPT1=(CONSOLE,MFS-SCS1), OUTBUF=256
NAME	MAOP001
TERMINAL	NAME=MAOP002,COMPT1=(CONSOLE,MFS-SCS1), OUTBUF=256
NAME	MAOP002
TERMINAL	NAME=MAOP003,COMPT1=(CONSOLE,MFS-SCS1), OUTBUF=256
NAME	MAOP003
TERMINAL	NAME=MAOT001,COMPT1=(CONSOLE,MFS-SCS1), OUTBUF=256
NAME	MAOT001R
TERMINAL	NAME=MAOT002,COMPT1=(CONSOLE,MFS-SCS1), OUTBUF=256
NAME	MAOT002
TERMINAL	NAME=MAOT003,COMPT1=(CONSOLE,MFS-SCS1), OUTBUF=256
NAME	MAOT003

User Action

The following creates a session with IMST using LU MAOT001:

```
MAILOGON IMST
```

The following creates a cross-domain session with IMSP using LUMAOT002:

```
MAILOGON IMSP
```

The following creates a session with TSOT using LU MAOT003:

```
MAILOGON TSOT
```

Chapter 9: Working With System Image Definitions

Note: This chapter applies to all products except CA SOLVE:FTS.

This section contains the following topics:

Implement System Images (see page 110)
Access System Image Definitions (see page 111)
Define a System Image (see page 112)
Maintain System Images (see page 113)
Add Resource Definitions to a System Image (see page 113)
Why Do You Need to Define Resources? (see page 114)
Access Resource Definitions (see page 114)
Define a Resource to a System Image (see page 115)
Time-outs (see page 137)
Specify Messages in a Resource Definition (see page 138)
Extend the Definition of Resource Message Rules (see page 141)
Execute an NCL Procedure (see page 146)
Create User-defined Resource Subclasses (see page 147)
Logical Resources (see page 148)
Define Resource Relationships (see page 148)
Staged Image Load and Shutdown (see page 152)
System Load Balancing (see page 152)
Resource Definition Maintenance (see page 155)

Implement System Images

You define the operations requirements of the resources to be managed on a system in a system image. You must create a system image definition before you can define the resources you want to manage.

Note: For information about creating your initial system image, see the *Administration Guide.*

You define a system image by giving it a name and assigning a version number. You must also nominate a home system on which the image can be loaded.

Note: If you are defining a system image for a subordinate region in a multisystem environment, the image name is restricted to that specified during the linking operation.

You assign different version numbers to create different views of the managed resources in the knowledge base. For example, the current live version of an image is version 0001, the new version (which contains changes that are not yet implemented) is version 0002, and the old version (a view of the image as it was before the changes in the current version were implemented) is version 0003.

You can also use the version to reflect the creation date of the system image, for example, 0705 for May 2007.

Note: The $TEMPLAT system image name is reserved for templates. For more information about how to work with templates, see the *Administration Guide.*

Example

You are responsible for the IS department of the eastern branch of a company, and you have defined a development system image called EASTTEST 0001. When the branch needs to add or change some definitions, you copy EASTTEST 0001 to a new image called EASTTEST 0002. You change the contents of EASTTEST 0002 and test it until it is working properly.

When the changes are working, you copy EASTTEST 0001 to a new image called EASTTEST 0003. This new image can be used as a backup if EASTTEST 0002 fails for any reason. You then load EASTTEST 0002 to make it temporarily the current image and delete EASTTEST 0001. You can then copy EASTTEST 0002 to a new EASTTEST 0001 and load this EASTTEST 0001 image.

The next time you want to update the current system, you update EASTTEST 0002, test it and back up EASTTEST 0001 to EASTTEST 0003. You then proceed as before to create a new EASTTEST 0001 by copying the tested EASTTEST 0002.

Access System Image Definitions

The knowledge base contains a list of system image definitions.

To access the list of system image definitions stored in the knowledge base

1. Enter the **/RADMIN.I** at the prompt.

   The system image definitions appear.

   Active system images have the following color coding:

   - White is used to indicate the active system image used by the local region.
   - Turquoise is used to indicate the active system images used by connected regions.

Note: Updating an *active* system image has an immediate effect on the operations of the resources controlled by the region.

Define a System Image

You must add a system image definition to the knowledge base.

To add a system image definition

1. Press F4 (Add) from the System Image List panel.

   A System Image Definition panel appears. You can now define the system image.

Example

In this example, the system image is called EASTTEST, which represents a development system of the same name (the home system). The following panel shows the completed definition:

```
PROD------------- ResourceView : System Image Definition ---------------------
Command ===>                                                  Function=ADD


System Name ......... EASTTEST
Database Version .... 0001
Home System ......... EASTTEST  (...where the image will load. Blank for ALL)

Short Description ... Development system for Eastern
Long Description .... Handles application development and testing by Eastern
                      branch programmers.  Contains all the resources
                      necessary to maintain this environment.


EventView Ruleset to Activate +









 F1=Help      F2=Split     F3=File      F4=Save
                           F9=Swap                              F12=Cancel
```

For products that support event management, you can associate an EventView rule set with the system image by using the EventView Ruleset to Activate field. The rule set is activated when the system image becomes active, thus enabling event-based automation through the active event rules.

Note: By default, ResourceView and EventView rule actions are not executed in the MANUAL operation mode. However, the actions can be enabled by using the Perform Action in Manual Mode? field of the AUTOIDS Customizer parameter group.

Maintain System Images

You can browse and update system image definitions, and copy and delete system images, from the System Image List panel.

Notes:

- If you make changes to the definitions belonging to an active system image, the changes become effective in the active image immediately.
- In a multisystem environment, this active image can be in one of the connected regions.

Merge System Images

You can use the C (Copy) action code to merge two system images. When you merge the images, you can specify whether you want to overlay existing definitions. The target system image contains the merged definitions.

To merge a source system image into a target system image

1. From the System Image List panel, enter **C** beside the source system image.

   The system image definition appears.

2. Complete the following fields:

   System Name

   Specifies the name of the target system image.

   Database Version

   Specifies the version number of the target system image.

3. Press F3 (File) to initiate the merging operation.

   You are prompted to confirm the operation.

4. Indicate whether you want to overlay definitions that already exist in the target. Press F6 (Confirm) to start the actual merging operation.

   The system images are merged. A panel is displayed to indicate the progress of the operation.

Add Resource Definitions to a System Image

Resource definitions are stored in the knowledge base in a system image.

Why Do You Need to Define Resources?

To enable the region to manage your resources, you need to define those resources to the knowledge base by using resource definitions.

The resource definitions belong to system images. To work with the definitions, you should have defined at least one image.

Access Resource Definitions

You can access resource definitions through one of the following:

- Resource Definition panel
- System Image List
- Status Monitor

Access Resource Definitions from the Resource Definition Panel

To access resource definitions from the Resource Definition panel

1. Enter **/RADMIN.R.***resource-class-name* at the prompt.

   The resource list appears.

   Note: The list panel identifies the system image to which the resources belong. If you want to list the resources belonging to another system image, update the System Name and Version fields.

Example

To list started tasks

1. Enter **/RADMIN.R.STC** at the prompt.

   The list of started tasks appears.

Access Resource Definitions from the System Image List

To access resource definitions from the System Image List

1. Enter **/RADMIN.I** at the prompt.

   The System Image List appears.

2. Enter **R** beside the appropriate system image.

   The resource definitions in that image appear.

Access Resource Definitions from the Status Monitor

The Status Monitor displays the resources that are being monitored. You can update (DB line command), copy (CPY line command), and add (F4 function key) resources form the monitor.

To access resource definitions from the Status Monitor, enter the shortcut to the monitor.

The monitor appears listing the monitored resources.

Define a Resource to a System Image

To define a resource, you must enter information on a series of panels. Each product supports its own resource classes and these classes all have a different series of panels.

You can use variables (see page 386) as data in a resource definition.

To define a resource to the system image

1. Press F4 (Add) from the resource list window.

   A General Description panel appears. This is the first in a series of panels you use to define the resource to the system image.

Notes:

- You can facilitate the definition of local resources to a system image by using AutoAssist facilities (if supported by your product).
- The class INTNL is for internal use only. Do not add resources of this class.

Describe the Resource

Use the General Description panel to specify the resource name and type.

Use Resource Templates

A set of sample templates for resources is supplied with most resource classes. You can use these templates or define your own templates to simplify the task of creating resource definitions.

You apply a template using the options under Template Selection on the General Description panel.

You can specify how you want to handle any map or processes specified in the template in the Copy Map and Copy Process fields. Valid values are as follows:

No

Do not copy.

Replace

Copy; replace if the map or process exists in the system image.

Yes

Copy only if the map or process does not exist in the system image.

Note: Global processes are already visible to the resource definition and are not copied into the system image, irrespective of the value in the Copy Process field.

In the Template Selection window, you can perform a number of actions by entering the action code beside the TemplateName field. These actions are as follows:

B (Browse)

Displays the contents of a template definition.

L (List)

Displays the list of available templates.

M (Merge)

Merges the values in a template with the existing values in a resource definition (for example, to update a resource definition). Merging does not overwrite existing values, so ensure that you delete the values in the resource definition that are to be replaced. Merging also does not set data that conflicts with existing data (for example, a process name is not set when a command already exists).

O (Override)

Populates the fields in a resource definition with the values in a template (for example, to add a resource definition by using a template as the model).

R (Reset)

Clears the fields in a resource definition, then overrides it with the values in a template.

Note: The M, O, and R actions do not affect the system image, resource name, and description fields specified in a resource definition. Before you apply the M, O, or R action, you should ensure that the resource name is specified.

Specify the Operation Mode

The operation mode specifies how the defined resource operates. During operation, the mode specified can be restricted by the global operation mode of the system image.

The global operation mode can be either AUTOMATED or MANUAL, with the former having a higher rank. If the global operation mode is MANUAL, the resource operation mode of AUTOMATED is forced to MANUAL during operation.

Note: For information about how to set the global operation mode, see the *Administration Guide*.

To specify the operation mode, specify AUTOMATED, OFF, MANUAL, or IGNORED in the Operation Mode field.

Control the VTAM ACB of a Resource

Some resources have ACBs. The ACBs must be available before these resources can start successfully. The ACB Name field enables you to invoke the following actions automatically during resource startup and shutdown:

- The ACB is activated by the following command before the resource is started:

  `VARY NET,ACT,ID=`*`acb-name`*

- The ACB is inactivated by the following command if the resource is stopped successfully:

  `VARY NET,INACT,ID=`*`acb-name`*`,IMMED`

To control the VTAM ACB of a resource, identify the ACB in the ACB Name field.

Define the Availability of the Resource

Some resource classes let you specify changes to the normal availability of the resources by using availability maps. An availability map also enables you to schedule changes to the operation mode and the starting of processes. You can attach more than one resource to an availability map.

To attach the resource to an availability map, select the map in the Availability Map field.

If no suitable map is available, you can add a map through the F10 (EditMap) function key.

Note: If you intend to form a parent-child relationship between resource definitions where the availability is determined by the parent, attach the map to the parent definition and define the desired state of the children to be always ACTIVE.

Note: The availability of a service overrides the availability of the resources that make up the service. If the resource always operates as part of a service, let the service handle the availability of the resource. Define the desired state of the resource to be always INACTIVE. When the service starts, it places an ACTIVE desired state override on the resource.

Specify Monitoring Requirements

CA NetMaster NM for TCP/IP provides resource classes that let you control the monitoring of resource attributes.

To specify monitoring requirements, indicate whether you want to monitor attributes and (optionally) attach a map to schedule the monitoring.

Note: For more information, see the *Implementation Guide*.

Define the Activation Details

Some resource definitions let you specify the following activation details:

- How to activate the resource
- What to do if the activation is successful
- What to do if the activation is not successful

If you want to perform pre-activation processing, you can specify it on the State Change Exits panel.

To define the activation details for the resource

1. Press F8 (Forward) from the General Description panel.

   The Activation Details panel appears.

   The A (activate) command uses the information on this panel.

   Note: For information about how to use the A command, see the *User Guide*.

2. Specify the information required for starting the resource. For an example, see a template distributed with the product.

   - If the starting method is simple, specify a system command, time-out (see page 137) and status change information, and the expected completion message (see page 138).

     You can extend the message definition (see page 141).

   - If the starting method requires more than one command or requires the processing of multiple messages, use a process. You can create and maintain processes on the Process List panel, which is accessible through the /RADMIN.P panel path.

     Note: You can use the $NCL process to execute an NCL procedure (see page 146).

     If you use a process, your process must set the return code to one of the following values, to notify the region to set the correct actual state of the resource:

Return Code	Actual State
0	ACTIVE—process is successful.
12	Do nothing. For example, the process might have set the state already by using the SETSTATE macro on behalf of the region. **Note:** You should use this method only if the set state is acceptable to the ACTIVE desired state. The acceptable states are ACTIVE, DEGRADED, and STARTING. If the result of the process is unacceptable, set the return code to, for example, 8 to indicate that the process failed to achieve the desired state.
99	Proceed the same way as if a system command has been issued (that is, wait for the expected completion message and possibly perform time-out processing).

Return Code	Actual State
Other	Process is unsuccessful. If time-out processing is specified, perform it immediately. Do not wait for the period specified in the Timeout After field. If time-out processing is not specified, do *not* change the actual state. The region can proceed no further with the automated starting of the resource. The *automation status* is set to FAILED, and the logical state is forced to INERROR.

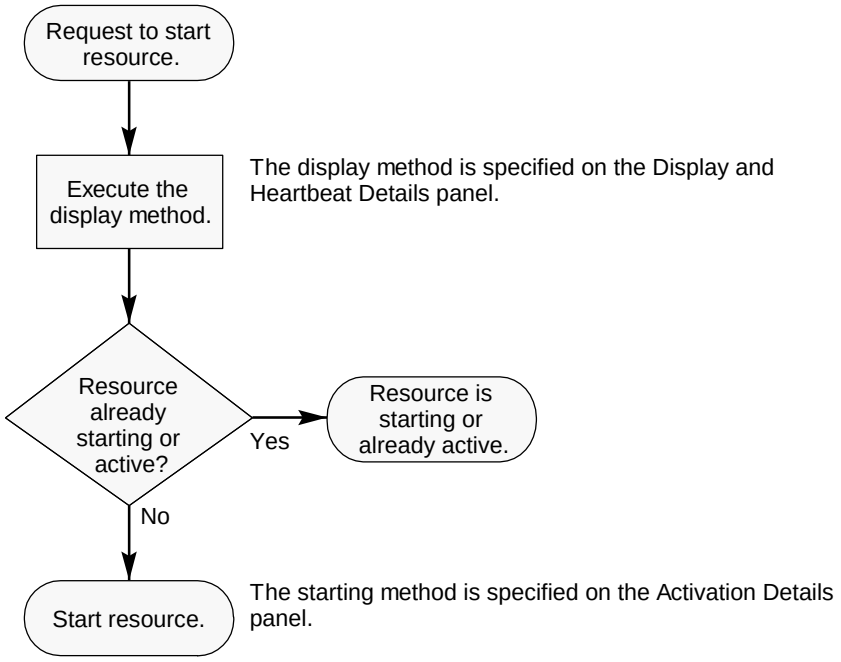
Note: If you provide *no* command or process for starting the resource, the region cannot start the resource when required. Instead of trying to start the resource, the region places the resource in the MANUAL operation mode. The exception is when the resource has a type of LOGICAL.

More information:

Logical Resources (see page 148)

How Activation Works

During automation, the region uses the display method to check the status of the resource that is to be started. If the resource is already starting or has already become active, the starting method is not executed. The following illustration shows the sequence of events that occurs during the starting operation:



Note: A similar sequence of events occurs when a request to stop a resource is processed.

How Restart Control Affects Resource Restarts

Conditions might arise when an automated resource becomes active momentarily only and then turns inactive, causing the region to restart the resource. If the condition persists, the region will continue to restart the resource every time the resource deviates from the ACTIVE state. To prevent this situation from continuing unchecked, you can use the restart control parameters to limit the number of times activation is retried.

The restart control parameters specify the maximum number of restarts that are permitted within a specified time period. The time period starts the first time the resource is activated.

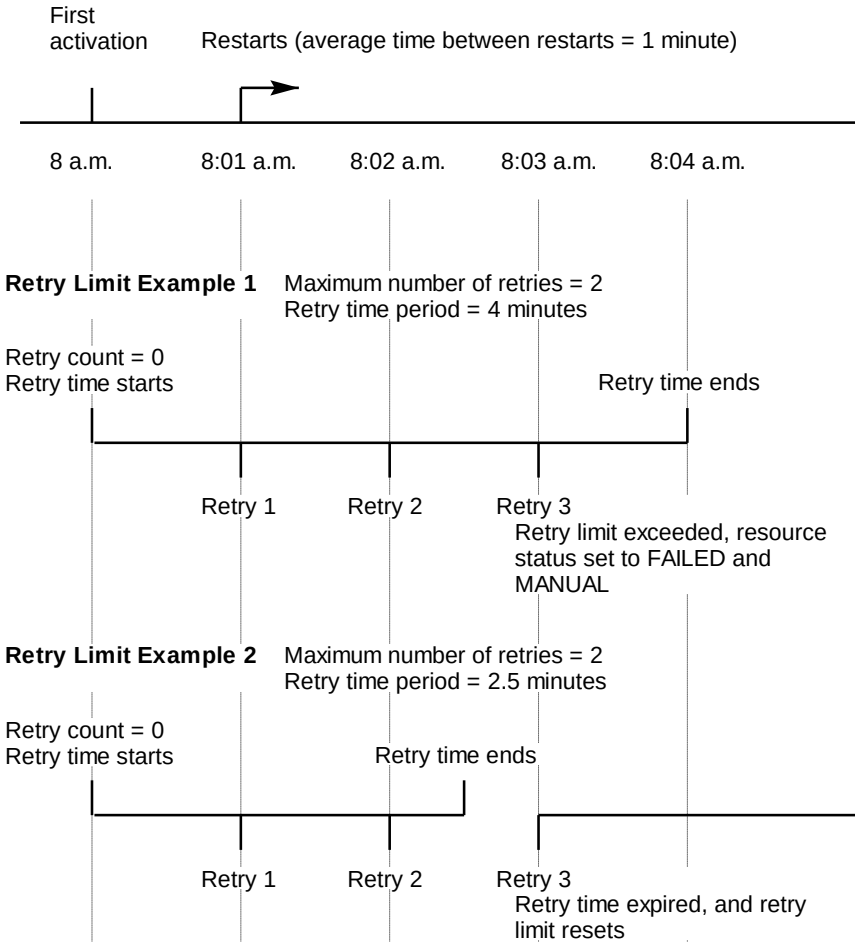
If this retry limit is exceeded, the actual state of the resource is set to FAILED and the operation mode of the resource is set to MANUAL. No further activation is performed.

If the retry limit is not exceeded, the next activation that occurs outside the retry time period will reset the retry count and period.

Example: Effect of Activation Limits on Resource Restarts

The following illustration shows how different retry limits affect resource restarts. In the first example, the retry limit is exceeded and the resource status is set to FAILED and MANUAL. In the second example, the retry limit is never exceeded and the restarts will continue unchecked.

Important! For restarts to be limited, the retry time period must be long enough for the retry limit to be exceeded.



If no restarts are required, the retry time period will expire and the parameters will have no effect on resource operations.

Effect of Restart Control on Manual Activation

Restart control applies irrespective of how the resource is activated, whether automatically or manually. The first activation resets the retry limit, and subsequent activations contribute to the retry limit.

Specify the Restart Control Parameters

You can specify the following restart control parameters:

- The retry limit
- What action to take if the retry limit is exceeded

To specify the restart control parameters

1. Enter **U** in the Restart Control Parameters window on the Activation Details panel.

   The Restart Control Parameters panel appears.

2. Complete the panel, and then press F3 (OK).

   Note: If you want to prevent the region from restarting the resource, enter **0** in the Retry Attempt Limit field and a time period in the Retry Time Limit field. If an attempt to restart the resource occurs within the specified time period, the activation is not performed and the resource state is set to FAILED. The operation mode of the resource is also set to MANUAL.

   The Activation Details panel reappears.

Define the Inactivation Details

Some resource classes let you specify the following inactivation details:

- How to inactivate the resource normally
- What to do if the inactivation is successful
- What to do if the inactivation is not successful

To define the inactivation details for the resource

1. Press F8 (Forward) from the Activation Details panel.

   The Inactivation Details panel appears.

   The T (terminate) command uses the information on this panel.

   Note: For information about how to use the T command, see the *User Guide*.

2. Specify the information required for stopping the resource. For an example, see a template distributed with the product.

   - If the stopping method is simple, specify a system command, time-out (see page 137) and status change information, and the expected completion message (see page 138).

     You can extend the message definition (see page 141).

   - If the stopping method is complex, use a process.

     Note: You can use the $NCL process to execute an NCL procedure (see page 146).

     If you use a process, your process must set the return code to one of the following values, to notify the region to set the correct actual state of the resource:

Return Code	Actual State
0	INACTIVE—process is successful.
12	Do nothing. For example, the process might have set the state already by using the SETSTATE macro. **Note:** You should use this method only if the set state is acceptable to the INACTIVE desired state. The acceptable states are INACTIVE and STOPPING. If the result of the process is unacceptable, set the return code to, for example, 8 to indicate that the process failed to achieve the desired state.
99	Proceed the same way as if a system command has been issued (that is, wait for the expected completion message and possibly perform time-out processing).
Other	Process is unsuccessful. If time-out processing is specified, perform it immediately. Do not wait for the period specified in the Timeout After field. If time-out processing is not specified, do *not* change the actual state. The region can proceed no further with the automated stopping of the resource. The *automation status* is set to FAILED, and the logical state is forced to INERROR.

Note: If you provide *no* command or process for stopping the resource, the region cannot stop the resource when required. Instead of trying to stop the resource, the region places the resource in the MANUAL operation mode. The exception is when the resource has a type of LOGICAL.

More information:

Logical Resources (see page 148)

How Inactivation Works

During automation, the region uses the display method to check whether the resource that is to be stopped is already inactive. If the resource is already inactive, the stopping method is not executed.

Define the Forced Inactivation Details

Some resource classes let you specify the following forced inactivation details:

- How to force inactivate the resource
- What to do if the forced inactivation is successful
- What to do if the forced inactivation is not successful

To define the forced inactivation details for the resource

1. Press F8 (Forward) from the Inactivation Details panel.

   The Force Inactivation Details panel appears.

   The TF (terminate by force) command uses the information on this panel. The T (terminate) command might also use this information if a time-out action of Try Force Inactivation is specified on the Inactivation Details panel.

   Note: For information about how to use the TF and T commands, see the *User Guide*.

2. Specify the information required to force the resource to stop. For an example, see a template distributed with the product.

   - If the forced method is simple, specify a system command, time-out (see page 137) and status change information, and the expected completion message (see page 138).

     You can extend the message definition (see page 141).

- If the method is complex, use a process.

  Note: You can use the $NCL process to execute an NCL procedure (see page 146).

  If you use a process, your process must set the return code to one of the following values, to notify the region to set the correct actual state of the resource:

Return Code	Actual State
0	INACTIVE—process is successful.
12	Do nothing. For example, the process might have set the state already by using the SETSTATE macro. **Note:** You should use this method only if the set state is acceptable to the INACTIVE desired state. The acceptable states are INACTIVE and STOPPING. If the result of the process is unacceptable, set the return code to, for example, 8 to indicate that the process failed to achieve the desired state.
99	Proceed the same way as if a system command has been issued (that is, wait for the expected completion message and possibly perform time-out processing).
Other	Process is unsuccessful. If time-out processing is specified, perform it immediately. Do not wait for the period specified in the Timeout After field. If time-out processing is not specified, do *not* change the actual state. The region can proceed no further with the automated forced stopping of the resource. The *automation status* is set to FAILED, and the logical state is forced to INERROR.

Note: If you provide *no* command or process to force the stopping of the resource, the region cannot force the resource to stop when required. Instead of trying to force the resource to stop, the region places the resource in the MANUAL operation mode. The exception is when the resource a type of LOGICAL.

More information:

Logical Resources (see page 148)

How Forced Inactivation Works

During automation, the region uses the display method to check whether the resource that is to be stopped by force is already inactive before executing the specified stopping method. If the resource is already inactive, the stopping method is not executed.

Define the Display and Heartbeat Details

Some resource classes let you specify how to find out about the status of the resource. Before the region starts or stops a resource, the region executes the display method to check the actual state of the resource. The heartbeat interval enables you to set up regular checking of the resource status.

To define the display and heartbeat details for the resource

1. Press F8 (Forward) from the Force Inactivation Details panel.

   The Display and Heartbeat Details panel appears. For an example, see a template distributed with the product.

   Generally, the CHECKALL, CHK (check), and D (display) commands use the method on this panel when checking or displaying the resource status. The exceptions are resources that have their own special D commands. You can identify these exceptions from the list of command definitions. To access the list, enter the **/ASADMIN.C** path.

2. Specify the display method required for status checking. You can specify a system command or a process without specifying the heartbeat interval.

   Important! The heartbeat feature increases CPU consumption. You should use this feature only for resources that are liable to change state without an accompanying message.

   - If the display method is simple, specify a system command, heartbeat interval, and the expected response message (see page 138).

     You can extend the message definition (see page 141).

   - If the display method is complex, use a process.

     Note: You can use the $NCL process to execute an NCL procedure (see page 146).

     If you use a process, your process must set the return code to one of the following values, to notify the region to set the correct actual state of the resource:

Return Code	Actual State
0	ACTIVE.
1	STARTING.
2	STOPPING.
3	DEGRADED.
5	INACTIVE.
6	FAILED.
12	Do nothing. For example, the process might have set the state already by using the SETSTATE macro.
Other	UNKNOWN.

Define the Status Monitor Message Details

The status monitor message details specify rules that can be triggered by messages to perform actions.

From the Status Monitor Messages panel, you can set up message rules to perform actions that are message dependent. The information defined on this panel includes the message identifier and text, the message rule priority, and the effect of the rule on the resource state. You can define up to 97 lines of message rules.

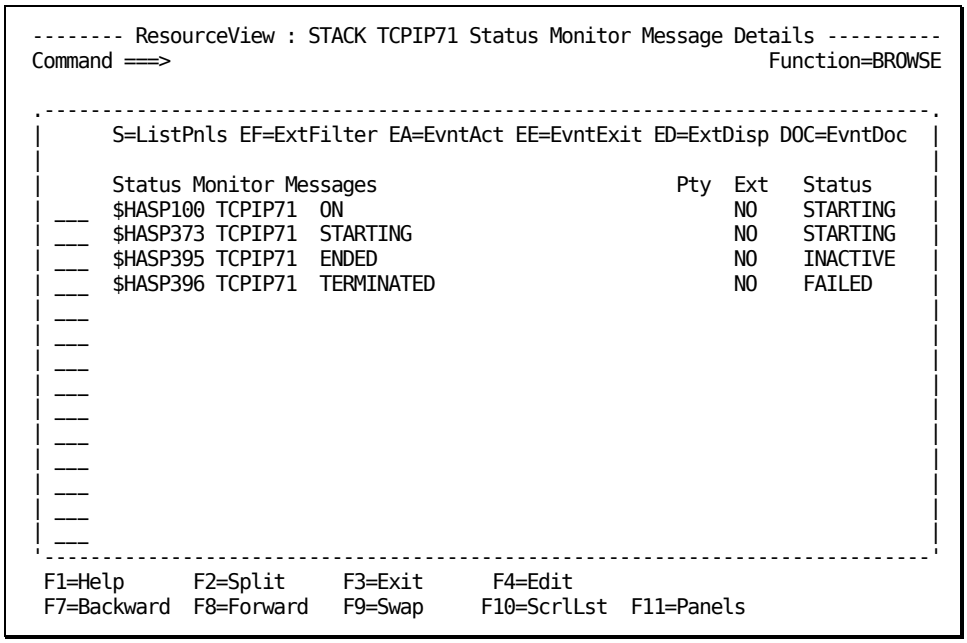
The rule priority is only significant when rules overlap. Priorities are organized in descending numeric order. For example, the value 10 indicates a higher priority than the value 20. If rules overlap, only the rule with the highest priority is processed. If overlapping rules have the same priority value, the most specific rule has the highest priority. For a set of overlapping rules, the rules are sorted in the order of decreasing priority.

The modifiable Status fields are optional. If a Status field is blank, the message rule does not trigger a state change.

The region uses the rules to monitor the messages for the particular resource. If a rule is triggered, the resource state is updated according to the value in the Status column.

Example

The following shows the Status Monitor Message Details panel for a stack resource.



Extended Message Definition Considerations

Normally, actions associated with the extended message definition are performed only if the resource is in the AUTOMATED operation mode. This condition is set in the AUTOIDS Customizer parameter group. To access the list of parameter groups, enter the **/PARMS** shortcut.

The information specified on the Define Extended Display Attribute panel, however, is always acted on irrespective of the operation mode.

Use the RECOVERED Actual State

The RECOVERED actual state enables the recognition of messages that are the results of recovery from a DEGRADED or FAILED state.

DEGRADED and FAILED are special actual states consisting of a base state and an applied flag as follows:

Actual State	Flag	Base Actual State When the Flag Is Applied
DEGRADED	DEGRADED	ACTIVE
FAILED	FAILED	Previous actual state

Note: The base actual state can be changed by the automation engine.

To detect a recovery condition, you should use the RECOVERED actual state. When the state is triggered, it removes the DEGRADED or FAILED flag, and sets the base state to ACTIVE.

Desired State Management Considerations

If your product uses desired state management or your resources are running in AUTOMATED mode, the automation engine acts as follows when the actual state of a resource changes to an actual state that is not the same as its desired state:

If the actual state changes to ...	The automation engine ...
ACTIVE or INACTIVE	Attempts to bring the resource to the desired state.
DEGRADED	Attempts to bring the base state of the resource to match that of the desired state.
FAILED	Does not attempt recovery unless recovery actions are specified in the message rule.
STARTING or STOPPING	Takes no action unless actions are specified in the message rule.
UNKNOWN	Invokes the specified display processing to attempt to determine the state. However, if the UNKNOWN state is the result of an activation or inactivation time-out, the engine sets the automation status to FAILED.

Implement the State Change Exits

Some resource classes let you specify state change exits through processes. You can specify two types of exit processes:

- A process that executes before the starting method specified on the Activation Details panel is performed.
- Processes that execute when specified state changes occur. For example, if a resource fails, you may want to invoke a process that writes a problem report. You can specify a process to execute when the actual state, the desired state, or the logical state of the resource changes.

To implement the state change exits for the resource

1. Press F8 (Forward) from the Status Monitor Message Details panel.

   The State Change Exits panel appears.

2. Specify the required exits.

   The specified information is saved when you save the resource definition.

Example: Failure Exit

The following panel shows an example in which the State Change Exits panel specifies that a process be executed when the resource fails:

```
. State Change Exits ------------------------------------------------------.
|                        S/B=Browse U=Update I=Interactive L=List P=Parameters |
|     Invoked                      Process  Parameters                       |
| ___ BEFORE ACTIVATION                                                      |
|     State   Change    Change                                               |
|     Type    From      To         Process  Parameters                       |
| ___ ACTUAL  ANY       FAILED     PROBSOLV ACB=PROB                         |
```

Define the Logging Details

The Automation Log Details Panel displays information about the size of the temporary log for the resource (called a *transient log*), the destination of the logged information, and the type of information logged.

If your product region uses too much storage, tune it by reducing the size of your transient logs. Transient logs can use a lot of storage, and over the lifetime of a product region, can grow until they reach their maximum size. Set the Log Table to the appropriate size according to the resource you want to monitor. For example, you do not need 9,999 lines for a JES initiator or any other resource. Log All System Messages and Log Internal Audit Trail should only be enabled when debugging a resource.

This product supports the STL action against images listed at /RADMIN.I.L and the SETTLOG command at status monitors, which lets you update the transient log size globally.

Note: For more information, see the *Administration Guide*.

The following shows the Automation Log Details panel:

```
Resource Log Controls

  Transient Log Size ......... 0150
  Log to Automation Log ...... YES
  Log to Console ............. NO
  Log to OCS Window .......... NO

Resource Log Content Controls

  Log All System Msgs ........ NO
  Log Internal Audit Trail ... NO
```

Transient Log Size

Specifies the total number of messages that the transient log can hold. The default is set in the AUTOTABLES Customizer parameter group during the initialization of the region.

The transient log is kept in memory. A large size can affect extended memory allocation.

The transient log is cyclic. That is, if the log is full, a new message displaces the oldest message.

Limits: 0 through 9999

Log to Automation Log

Specifies whether to write transient log messages to the activity log.

Default: NO

Log to Console

Specifies whether to send messages to the system console as WTO messages. Select this option only for resources that are started before VTAM and if you want to debug the resource definitions.

Default: NO

Log to OCS Window

Specifies whether to send messages to users who are monitoring from OCS.

Default: NO

Log All System Msgs

Specifies whether to log all messages for a resource. If the value is NO, the region logs only messages that match the message rules in the resource definition.

Messages are written to the transient log and to any destinations specified in the previous options.

Default: NO

Log Internal Audit Trail

Specifies whether to log the internal audit trail. The audit trail is a detailed log of the processing and actions performed in the region, and is useful for debugging the resource definition.

Note: The option creates a high volume of message flow. Turn on the option only if required.

The audit trail is logged to the transient log and any destinations specified in the previous options.

Default: NO

Specify the Owner Details

The Owner Details panel lets you identify up to two people who can be contacted if the resource has operational problems. The template does not assign this information.

To specify the owner details

1. From the Automation Log Details panel, press F8 (Forward).

   The Owner Details panel appears.

2. Specify the details, as required.

   Press F1 (Help) for information about the fields.

Implement the Extended Function Exit

Some resource classes let you provide additional operator functions using an NCL procedure. The procedure is invoked when an operator issues the XF command against the resource.

To implement the extended function exit for the resource

1. Press F8 (Forward) from the Owner Details panel.

   The Extended Function Exit panel appears.

   Specify the exit NCL procedure that provides the extended functions. This procedure has access to variables with the prefix ZRM.

   The specified information is saved when you save the resource definition.

More information:

Variables Available to a Command NCL Procedure (see page 176)
Knowledge Base Variables (see page 386)
Status Variables (see page 390)

Time-outs

You can specify how long to wait for the response to an action in the Timeout After field on the following panels:

- Activation Details panel
- Inactivation Details panel
- Force Inactivation Details panel

Usually, the response to the action sets the actual state of the resource to the state specified in the Status column.

However, if for any reason the expected response to the action does not come, then, depending on the timeout settings, the following occurs:

- If you have *not* specified a value in the Timeout After field, the region keeps waiting for a response until a significant event changes the status of the resource.

  The exception is the time-out for a resource of the LOGICAL type. For a logical resource, the default value for the Timeout After field is 0 (that is, no waiting).

- If you specified a value in the Timeout After field and you specified the action to take when time-out occurs, the region takes the specified action. The action sets the resource to a specified actual state. On the Inactivation Details panel, you can, instead of specifying the actual state to set, specify that the resource be stopped by force according to the forced inactivation details.

- If you specified a value in the Timeout After field but you did *not* specify any action, the region performs the display processing specified on the Display and Heartbeat panel.

  Note: For a logical resource, display processing is not applicable. The actual state of the logical resource is set directly to the expected result of the action. A starting action results in an actual state of ACTIVE; a stopping action results in an actual state of INACTIVE.

  If the response to the action indicates that the desired state is satisfied, the region sets the correct actual state.

  If the response to the action indicates a state other than the desired state, the region can proceed no further with the automated process. The region does *not* change the actual state of the resource, but sets the *automation status* for the resource to FAILED and forces the logical state of the resource to INERROR.

Specify Messages in a Resource Definition

While defining a resource, you may need to specify messages on the following panels (if they apply to your resource class):

- Activation Details panel
- Inactivation Details panel
- Force Inactivation Details panel
- Display and Heartbeat Details panel
- Status Monitor Message Details panel

Enter the following querying codes in a message field to help you specify the messages:

- Enter **?** to obtain a list of suggested messages.
- Enter **??** to obtain a list of all the messages learned from EventView, if present.
- Enter **???** to obtain a list of the messages in the transient log for the resource. This feature simplifies message selection, because a transient log contains only messages that are relevant to the resource.

Select Messages in the Transient Log

When you are working on a resource definition, you can use the transient log of a resource as a source of resource-specific messages.

Before you use the ??? action, you should ensure the following:

- On the Automation Log Details panel of the resource definition, the value in the Log All System Msgs field is YES.
- The definition has been saved in the knowledge base.
- The system image that contains the resource is active.

To select messages in the transient log for the resource

1. Enter the **/RMON** shortcut.

   The status monitor appears. If you are using a filter, you might need to change the filter in order to see the resource. (Use the FILTER command to list the defined filters.)

2. (For resources that have specified activation and inactivation methods) Find the resource, ensure that the operation mode is IGNORED or MANUAL, and then issue the **A** (activate) and the **T** (terminate) commands manually.

   The resource starts and stops.

3. Issue the **D** (display) command when the resource becomes active and when the resource becomes inactive.

   These actions populate the transient log with messages.

4. Enter **DB** beside the resource.

   The Panel Display List panel for the resource definition appears.

5. Select the required panel, and enter **???** in a message field on the panel.

   The Transient Log Browse panel appears.

6. Browse the transient log, and select the appropriate message.

   The Resource Definition panel is redisplayed with the selected message in the message field.

Special Message Prefixes

Normally, a resource definition in an active system image recognizes only messages for the corresponding resource. However, the following special message prefixes are available to enable a resource definition to handle special types of local messages. You *cannot* combine prefixes.

$AA-

A resource definition can generate a user-defined resource event. For another resource definition to recognize the event message, you must prefix the message by $AA- when you specify the message rule in that definition.

For example, ALERT - VTAM AVAILABLE is specified as the event message on the Define Event Related Action panel of a resource definition. To enable the resource you are defining to recognize that message, you must specify $AA-ALERT - VTAM AVAILABLE as the message text to look for.

$DN-

A message can be a delete operator message (DOM) notification. For a resource definition to recognize a DOM notification, you must prefix the message by $DN-.

For example, specifying $DN-IEF233A M 380 enables a resource definition to recognize the DOM notification for the tape mounting request message IEF233A M 380.

$MN-

A resource definition recognizes messages that relate to the resource itself only. For a resource definition to recognize messages from other resources and from the operating system, you must prefix the messages by $MN-.

For example, you might want to detect the starting and the stopping of a batch job that performs system backup. Detecting those messages enables you to stop and restart a resource, as required, by changing the desired state of the resource through the SVCMD macro in a process.

Extend the Definition of Resource Message Rules

You can extend the definition of any message rule in a resource definition (as entered on the Activation, Inactivation, Force Inactivation, Display and Heartbeat, and Status Monitor Message Details panels).

To extend a rule

1. Enter **S** beside the message.

   A selection panel appears, listing the following extended message definition panels:

   - Define Extended Filter panel
   - Define Event Related Actions panel
   - Define Event Exits panel
   - Define Event Display Attribute panel
   - Event Documentation panel

Define an Extended Message Filter

Use the Extended Filter Definition panel to specify the criteria for analyzing message text. A message is acceptable if it meets the criteria you specify here. For example, the same message may indicate different resource states, depending on the event that generates the message.

Note: If you want to capture a message that has leading blanks, you do not need to specify the leading blanks on the message filter panel. However, on the Extended Message Filter panel, absolute position is important so leading blanks must be counted when using start position of text.

Define Event-related Actions

Use the Define Event Related Actions panel to define particular actions that the region performs when it receives a particular message. The following actions are available:

- Change the operation mode of the resource
- Issue a system command, reply to a WTOR message, or generate a resource event—use the $AA- message prefix when specifying a message rule in a resource definition if you want the definition to recognize resource events
- Log a user-defined message
- Execute a process

Note: You can use the $NCL process to execute an NCL procedure (see page 146).

Implement Event Exits

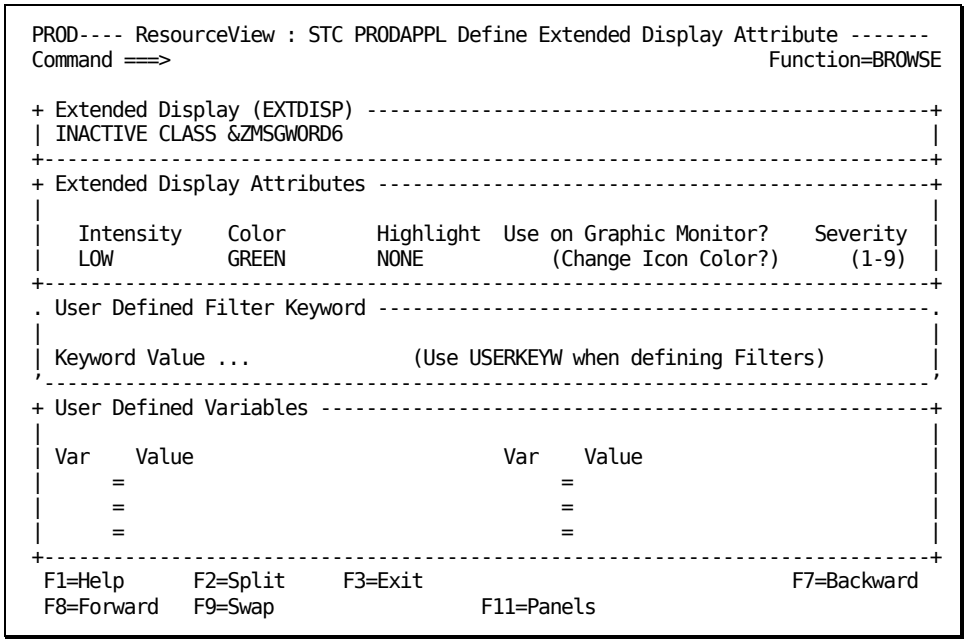
Use the Define Event Exits panel to define state change exits, problem exits, or other exits that are specific to this message rule. These exits are invoked when a message satisfying the rule is received. For example, if the rule is for a message generated by an event that causes the performance of the resource to degrade, you may want to specify an exit that logs the degradation in a problem management application.

Define Extended Display Attributes

Use the Define Extended Display Attribute panel to do the following:

- Override the status monitor fields.
- Override the display attributes.
- Define keyword that can be used by status monitor filters.
- Define information that can be displayed on icons.

The following is an example of the Define Extended Display Attribute panel. If triggered, it provides the specified extended display in green, where &ZMSGWORD6 is the sixth word of the triggering message.



Override the Status Monitor Fields

Use the Extended Display field to define the text to be overlaid on the status monitor line for this resource. The override text can pick up specified words from the original message if you specify &ZMSGWORD*n*. For example, if you specify an extended display of ACTIVE CLASS=&ZMSGWORD9, the override text displays the class as the ninth word in the original message.

If you leave this field blank and the rule triggers a state change, the status monitor line resets to normal display. If the rule does not trigger a state change and you want the monitor line reset to normal display, specify ##RESET## in the field.

You can use the Severity field in the Extended Display Attributes window to determine whether an extended display can be overwritten by another extended display. When an extended display is invoked, it can overwrite a display of equal or lesser severity only. The field ensures that the monitor displays the most severe condition. The greater the value in this field, the lesser the severity. For example, 9 is less severe than 1.

An operator, using the status monitor, can enter the primary command EXTDISP ON to display override text on the monitor or EXTDISP OFF to display standard message text. The default is to display override text.

Override the Display Attributes

Use the Extended Display Attributes window to specify the display attributes and how an icon on the graphical monitor is affected by the attributes.

- Normally, the display attributes of a status line on a status monitor are determined by the logical state of the relevant resource. By using the fields in this window, you can override the normal display attributes.
- An icon on the graphical monitor displays the status of the resource that is in the worst logical state. The Use on Graphic Monitor? field enables you to indicate whether the specified extended display color should be used on an icon.

  If you do not want the icon to use the specified extended display color, specify **NO** (the default) in the Use on Graphic Monitor? field. For example, if the resource does not require attention, transferring the color to the icon can cause an undesirable change in the icon color.

  If you want the icon to use the specified extended display color, specify **YES** in the Use on Graphic Monitor? field. This setting also forces the state ranking of the resource to equal that of the FAILED logical state. The icon can thus be forced to display this resource even though it is not in the worst logical state. For example, if you rate the condition of a tape mount request very important, specify **YES** in the Use on Graphic Monitor? field.

Use the User-defined Keyword

The keyword is used by status monitor filters to determine whether to display the status of a resource on the basis of the processed message.

For example, you may want to suppress the display of a resource on the status monitor when a particular message is detected. To do this, specify a keyword in the rule for that message and use the keyword as a criterion in a status monitor filter to *not* display the resource. When an operator uses the filter to view the resources on the status monitor, the resource is normally displayed on the monitor. However, if a message arrives and satisfies this message rule, the resource disappears from the monitor because the keyword in the filter tells the monitor *not* to display the resource.

Use User-defined Variables

Use the User Defined Variables window to specify data that can be used within an icon definition. The data is only available when the resource for which the data is defined has the worst status of all resources within an icon.

Document the Message Rule

Use the Event Documentation panel to record information about a message rule. For example, you may want to document what causes a particular ABEND message.

Execute an NCL Procedure

$NCL is a special process definition that is applicable across all defined system images. The $NCL process enables you to execute an existing NCL procedure directly. You do *not* need to define a process to execute an NCL procedure. The $NCL process return code inherits the return code set by the NCL procedure.

The command has the following format:

```
$NCL $NCL=ncl-procedure-name
parameter-1=value-1
.
.
.
parameter-n=value-n
```

$NCL=*ncl-procedure-name*

Names the NCL procedure. The name is mandatory.

parameter-1=value-1 ... parameter-n=value-n

Supplies parameter values to the NCL procedure. The parameters must be in keyword format (for example, NAME=*resource-name*). The parameter names must not start with $.

Note: You cannot use the $NCL process from within another process. To do this, use the EXECNCL macro.

Create User-defined Resource Subclasses

Note: This section applies only to CA NetMaster FTM, CA NetMaster NA, CA SOLVE:Operations Automation, and CA SOLVE:Operations Automation for CICS.

User-defined resource subclasses enable you to categorize resources in a class defined by you. You use resource subclasses to categorize resources that do *not* belong to any of the supplied resource classes.

You can monitor and control the resources in the subclasses from the monitors. You can define resource templates for the resource subclasses.

You use the USRCLS resource class when you define resources for resource subclasses. The User Class field of the resource definition determines the resource subclass. A resource subclass is a resource type in the USRCLS resource class.

Initially, the User Class field has one valid value, LOGICAL (indicating a logical resource). To define resources for your own subclasses, you must first define the subclasses in the User Classes field prompt list. The name of a resource subclass can be up to eight characters long.

Note: When your status monitor displays resources of the USRCLS class, you do *not* see USRCLS as a resource class. The resource subclasses are displayed instead.

To add a new user subclass in the User Classes field prompt list

1. Enter **?** in the User Class field on the User Class Resource General Description panel.

   The User Classes field prompt list appears.

2. Press F10 (EditList).

   The list is updated.

3. Press F4 (Add).

   The Field Prompt Entry Definition panel appears.

4. Define the new user subclass, then press F3 (File).

   The subclass is added to the list.

Logical Resources

Note: This section applies only to CA NetMaster NA, CA SOLVE:Operations Automation, and CA SOLVE:Operations Automation for CICS.

The function of some resources is purely to elicit some sort of activity from another resource (for example, starting or stopping other resources). Other resources might be logical representations of groups of resources. Define these types of resources as the LOGICAL resource type.

Note: The name of a logical resource must contain alphanumeric, @, #, $, ., :, -, (, and) characters only. It must not be a number.

Logical resources respond to the actions of starting, stopping, and forced stopping differently from the other types of resources. Logical resources respond to these actions as follows:

- If no command or process exists for the action, the actual state of the logical resource is set to the expected result of the action. A starting action results in an ACTIVE actual state; a stopping action results in an INACTIVE actual state.
- If a command exists and no timeout processing is specified, the region issues the command and then *immediately sets the actual state of the logical resource to the value of the desired state*.
- If a command exists and timeout processing is specified, the timeout settings determine the final status of the resource. See Specifying Messages in a Resource Definition in this chapter for information about how to specify timeout behavior.
- If a process exists, the region acts the same way as for other resources (that is, the region responds according to the process return code).

Define Resource Relationships

The resources monitored by some products can have relationships. When you have defined those resources in a system image, you can specify the relationships between them. For each resource, you can specify two different relationships (see page 150) from it to other resources.

The region uses the relationship information during automated system startup and shutdown to determine the order in which to start or stop the defined resources.

You can define relationships between different types of resources in a system. The relationships between resources may be quite different for each system image definition held in the knowledge base.

There are two parties to a relationship: parent and child.

During automated operation, a parent must be active before dependent resources (its children) can be started. A child cannot be started automatically unless all its parents are active.

Similarly, a parent cannot be stopped automatically unless all its children are inactive.

You can use the GRT command to display resource relationships from the status monitor, the graphical monitor, and the System Image List.

To relate a resource to other resources

1. Enter **/RADMIN.R** at the prompt.

   The Resource Definition panel appears.

2. Enter **S** beside the resource class.

   The resource list appears.

3. Enter **R** beside the resource that you want to relate.

   The Existing Relationships List displays any relationships already defined for the resource.

4. Press F11 (Relate).

   A selection list of the resource classes appears.

5. Select the class of resource to which you want the selected resource to relate.

   A list of all resources of that class in the system appears.

6. Enter **C** or **P** beside a resource to define it as a child or parent.

   The updated list of resource relationships appears.

7. Press F3 (Exit).

   The Existing Relationships List appears.

Note: You can enter **R** beside a resource to display its other relationships. If you want to break a relationship with a resource, use the Unlink action codes.

Primary and Alternate Relationships

For each resource, you can define a primary relationship and an alternate relationship with other resources. During startup and normal operation, the region uses the primary relationship. During shutdown, the region uses the relationship specified in the OPSYSIDS parameter group. The default is the primary relationship. You can override this value when you issue the SHUTSYS and SHUTFORCE commands.

To access the list of parameter groups, enter the **/PARMS** shortcut.

Effect of Resource Relationships on Operations

Resource relationships affect systems operations in the following ways, depending on the resource operation modes:

Start a System That Contains Only Automated Resources

During system startup, the region starts all the resources without parents first, then the children, level by level, until all resources are started.

Resources are started subject to the availability requirements defined in the resource availability maps.

Stop a System That Contains Only Automated Resources

During system shutdown, the region stops its resources in an order that is the reverse of the startup sequence. The resources are stopped level by level up the relationship tree.

Start a System That Contains Automated and Manual Resources

The region starts resources in the same order, as when starting a system that contains only automated resources (see page 150). However, when it encounters a resource in the MANUAL or IGNORED operation mode, the region does not start the resource or any of its children, even if the children are in the AUTOMATED operation mode. The region starts all other automated resources. Manual resources are started manually. Once a manual resource becomes active, the starting sequence continues with its children automatically.

Automated resources are started subject to the specified availability requirements.

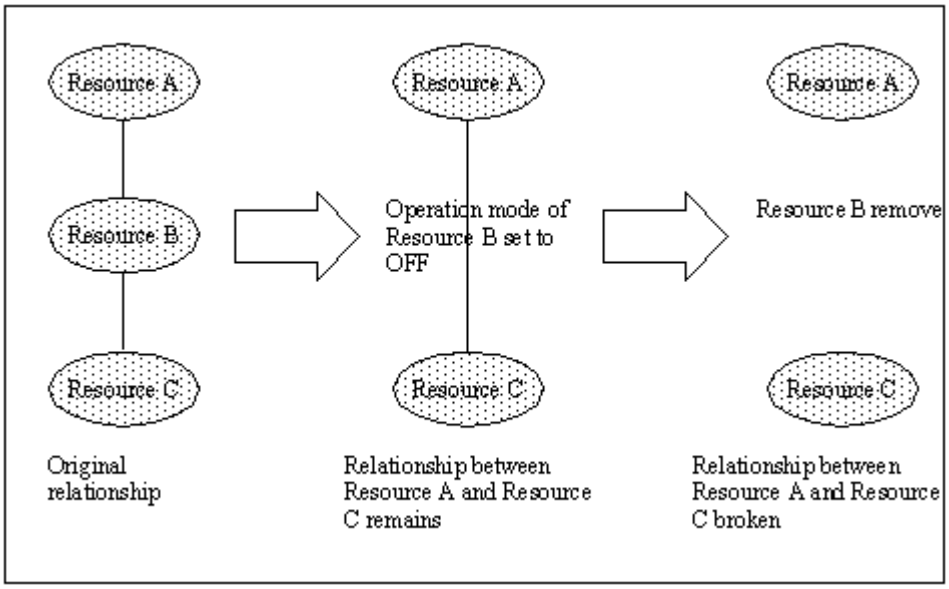
Stop a System That Contains Automated and Manual Resources

The region stops the resources in the same order as when stopping a system that contains only automated resources. However, when it encounters a resource in the MANUAL or IGNORED operation mode, the region does not stop the resource or any of its parents, even if the parents are in the automated operation mode. The region stops all other automated resources. Manual resources are stopped manually. When a manual resource becomes inactive, the stopping sequence continues with its parents.

Note: The SHUTFORCE command lets you stop all of the resources defined in an active system image, irrespective of the operation mode. Before stopping the resources, the command sets all MANUAL or IGNORED operation modes to AUTOMATED operation modes.

Effect of a Resource Set to the OFF Operation Mode on a Relationship

The following illustration shows the effect that a resource set to the OFF operation mode has on a relationship. As a comparison, the illustration also shows the effect of removing a resource definition from the relationship.



Staged Image Load and Shutdown

If you want to stage your image load or shutdown so that an operator can confirm whether to continue at strategic points in the process, create USRCLS resources in the shutdown and startup relationship chain that run a simple WTOR process as their activation and deactivation processing. Depending on the operator response to the WTOR, you can set the resource to inactive or active or take other action. Use the Timeout option to set a default action.

System Load Balancing

In a multisystem environment, you might want to balance the load on the systems by moving the control of certain shared resources from one system to another. The MV and MVT commands enable operators to perform that task. These commands operate on resources in local system images, and shared system images if your product supports them.

How MV and MVT Commands Move the Operation of a Resource

During a move operation, the following actions occur:

1. If the operation mode of the resource is AUTOMATED, it is set to MANUAL.
2. The resource is stopped or inactivated in the current system.
3. Control of the resource is transferred to the target system depending on whether the resource is in a local system image or shared system image.

For resources in local system images:

Note: For a successful move operation on a local resource, the resource must also be defined in the target system image.

1. The operation mode of the resource is set to OFF, removing the resource from the monitors. The current system no longer has control of the resource.
2. The resource is activated on the target system (unless it is INACTIVE and AUTOMATED). You can override this action, individually or for all resources.
3. The MANUAL override is removed, but not if the resource was INACTIVE and AUTOMATED, and you requested activation.
4. The operation mode of the definition in the target system image is set to the value before Step 1. The status of the resource is redisplayed under the target system image. The target system has control of the resource.

For resources in shared system images:

1. The resource is activated on the target system (unless it is INACTIVE and AUTOMATED). You can override this action, individually or for all resources.
2. The home system of the resource is changed to the target system. You can override this action, individually or for all resources.
3. The MANUAL override is removed, but not if the resource was INACTIVE and AUTOMATED, and you requested activation.

Define Shared Resources in Local System Images

For the MV and MVT commands to be successful, you should define the shared resources to the affected local system images.

Note: If your product supports shared system images, you do not need to perform this procedure.

To define shared resources

1. Define the required resources to the system image that normally controls those resources.

   The resources are defined for normal operation.

2. Duplicate the relevant resource definitions, resource relationships, availability map, and processes to the system images that you want to include in the set of load sharing systems.

   Note: The MV and MVT commands use the specified class and name in a definition to identify the resource. You do not need to duplicate the other information if you do not want to.

   The resources are defined to the load sharing systems.

3. Set the operation mode of the duplicate resource definitions to OFF.

   These definitions are initially dormant.

4. If a resource is part of a service, specify all systems for that resource in the service definition.

   The service is then able to track that resource as it is moved from system to system.

Resource Definition Maintenance

You can browse, update, copy, and delete resource definitions from the resource list.

Notes:

- Resources of the INTNL class with names in the form *xx*(*) are dynamic APPC resources that provide communications between regions. These resources are defined only for the duration of the APPC link. Do *not* change these resource definitions.
- If you only want to hide a resource definition from the region, set the operation mode or monitoring activity to OFF. The definition remains in the knowledge base but is not used.

When you copy a resource definition, the associated availability map and local processes are treated as follows if the map or process of the same name already exists in the target system image:

- For an availability map, you are prompted to indicate whether you want to replace the existing map.
- For a local process, the existing process is *not* replaced.

Chapter 10: Backing Up the RAMDB

Note: This chapter applies to all products except CA SOLVE:FTS.

This section contains the following topics:

Database Maintenance (see page 157)

Database Maintenance

The method you select for performing database maintenance depends on the configuration of your regions and your operations requirements.

Backup methods depend on whether your regions are:

- Unlinked, non-production regions
- Unlinked production regions
- Linked production and non-production regions

Unlinked Non-production Regions

To back up unlinked non-production regions

1. Stop the region.
2. Backup the database.
3. Restart the region.

Unlinked Production Regions

If the region cannot be shut down for database backup, we recommend that you back up unlinked production regions as described here.

To back up unlinked production regions

1. Create a duplicate region with its own knowledge base.

   Important! Ensure that the databases of the duplicate region are not on the same DASD as the production region databases.

   Note: If the region supports shared system images, create the duplicate on a different system. You cannot link regions that support shared system images if those regions are on the same system.

2. Use the Link Region and Synchronize Database option (on the Multi-System Support Menu) to link the duplicate region to the production region.

   The databases of the production region are copied to the duplicate region.

3. (Optional) In the duplicate region, switch off console consolidation in the CCONSOLIDATN parameter group (shortcut **/PARMS**).

4. Load an empty system image with no EventView rule set.

   Note: The duplicate region does not perform any automation, it just contains an up-to-date, mirror image of the production databases.

5. When backup is required, stop the duplicate region and perform the backup to tape.

Note: Any database updates that occur while the duplicate region is stopped are held by the staging file. The updates are sent to the duplicate region when it is restarted.

Linked Regions

Each region contains a copy of the database of each linked region. Updates to any database are immediately propagated to the databases of each linked region.

If backups to tape are required, and linked regions cannot be shut down, we recommend that you back up linked regions as described here.

To back up linked regions

1. Create a duplicate region, and link it to the production region.
2. Stop the duplicate region, and perform the backup to tape.

Note: Any database updates that occur while the duplicate region is stopped are held by the staging file. The updates are sent to the duplicate region when it is restarted.

Chapter 11: Reporting on Your RAMDB

Note: This chapter applies to all products except CA SOLVE:FTS.

This section contains the following topics:

About Reporting (see page 161)
Generate a Report (see page 162)

About Reporting

The reporting facility lets you search your knowledge base by specifying certain criteria to retrieve any definitions that match those criteria. For example, you may want to make a change to a macro and you need to find out what other parts of your system are affected by the change.

You can also update the retrieved definitions. For example, you may want to change all references to a macro. You can search for the macro and retrieve the relevant definitions, including the processes that reference the macro. You can then update those processes to refer to the required macro.

The following report formats that can be generated:

List

Provides a selection list of output in summary form.

Print

Prints detailed information.

Generate a Report

A report is generated by performing a search with specified criteria. To obtain a report, follow these steps:

1. Enter **/ASADMIN.R** at the prompt.

   The Reporting menu appears.

2. Enter *one* of the following at the prompt to specify the type of report that you want:

   - **B:** Displays the output as a selection list with summary information from which you can browse, update, or copy.
   - **P:** Prints the output with detailed information.

   The CAS: Lists panel appears.

3. Enter **S** beside the search criteria that you want.

   Each criteria selection displays a panel for you to specify search parameters. See the following sections for a description of how to specify the search parameters for the criteria you have chosen.

   Note: Enter **I** beside any of the listed criteria to get more information about the type of search it performs.

Search for Macros

If you selected the Search for Macro References criteria, the Search for Macro panel displays.

The macro search lets you retrieve macros that satisfy specified criteria.

To search for macros

1. Complete the following fields to specify one or more search parameters:

   Macro Name

   Specifies a macro name or a mask. For example, *VAR* specifies macro names that contain VAR.

   NCL Procedure

   Specifies the name of an NCL procedure or a mask. For example, $RMMC2* specifies NCL procedure names that begin with $RMMC2.

   Macro Description

   Specifies a string of text to search for in macro descriptions.

   Press F6 (Action).

   The output is printed or listed depending on what type of report you specified.

Search for Message Rules

If you selected the Search for Message Rules criteria, the Search for Message Rule panel displays.

The message rule search lets you locate all message rules that match the specified criteria.

Note: The fields do not support masks. For example, using IEC03* as the rule ID does not retrieve all rules with IDs that begin with IEC03. Instead, the criterion retrieves only those rules with message texts that have a first word of IEC03*. The asterisk (*) is interpreted as a real character, not a wild card.

To search for message rules

1. Specify one or more search parameters using the fields provided.

   The fields are a subset of the fields that are displayed when you define message rules. Press F1 (Help) for a description of the fields.

   Press F6 (Action).

   The output is printed or listed depending on what type of report you specified.

Search for Processes

If you selected the Search for Process References criteria, the Search for Process panel displays. The process search lets you locate all processes that match the specified criteria and the definitions in the knowledge base where the processes are referenced. Processes may be referenced in availability maps, resource definitions, or message rules.

To search for processes

1. Complete the following fields to specify one or more search parameters.

   System Name

   Specifies the name of a system image.

   Version

   Specifies the version of the system image.

   Process Name

   Specifies the name of a process or a mask. For example, DISP* specifies process names that begin with DISP.

   Process Description

   Specifies a string of text to search for in process descriptions.

   Note: We recommend that you specify a system name and version when performing the process search.

   Press F6 (Action).

   The output is listed or printed depending on the type of report you specified.

Use Your Search Output

If you have chosen to list the output of your search, you can browse, update, or copy any entry in the selection list. You can also issue the SORT and LOCATE commands to help you view and find information in the list.

If you have chosen to print your search output, you can hold the output and view it online before printing.

Note: For information about using the print services manager, see the *Administration Guide*.

Sort Your Output

To sort the output in the selection list

1. Enter **SORT ?** at the prompt.

   A list of sort criteria appears.

2. Enter the number of the sort criterion you want at the prompt.

   The search selection list with the SORT command and the criteria displayed at the command prompt appears.

3. Press Enter.

   The list is sorted.

Locate Output

You can use the LOCATE command to find specific entries in your search selection list. The command is performed on the field on which you sorted.

Example

To find the process $PROC01 in your search selection list

1. Enter **SORT $RM00NAME** at the prompt to sort the list by the object name.
2. Enter **L $PROC01** to locate the specified process.

Chapter 12: Defining Macros

Note: This chapter applies to all products except CA SOLVE:FTS.

This section contains the following topics:

Understanding Macros (see page 167)
NCL Procedures as Macros (see page 167)
Register and Maintain the Macros (see page 168)

Understanding Macros

A macro is an NCL procedure used to construct process steps. Sample macros are provided, but you can also write your own NCL procedures and use them as macros. Your NCL procedure must be manually distributed to linked regions. To enable the system to recognize an NCL procedure as a macro, you must register the macro.

For more information about using macros, see the *User Guide*.

NCL Procedures as Macros

A typical NCL procedure to be used as a macro must have the following features:

- Contain a brief description of the purpose of the procedure.
- Display panels that allow users to browse or change certain parameters used by the procedure. These parameters are displayed when a user applies the **P** (Parameters) action.
- Perform the required function when a process executes the macro.
- Return a code that a user can test to determine what the processing requirements are for later process steps.

The following macro template procedures are provided in CC2DEXEC for you to customize to suit your requirements:

- $RMMC00S, which executes macro actions
- $RMMC00D, which handles parameter definitions

Customize the $RMMC00S Macro Template

You need to set the value of #RMMACRO (that is, the macro name), then you need to customize the subroutines introduced by the following labels:

- SHORTDESC (optional)
- LONGDESC (optional)
- PARMHELP (optional)
- EXECMACRO (*mandatory*)

Further customization can be performed, but these are the recommended minimum changes to the $RMMC00S template.

Note: For examples of how this template is used, see other $RMMC*nn*S members in the CC2DEXEC data set.

Customize the $RMMC00D Macro Template

If you want parameter validation to work correctly, customize the subroutines introduced by the following labels:

- PINIT
- PSET
- PSAVE
- VMANDATORY

Further customization can be performed, but these are the recommended minimum changes to the $RMMC00D template.

Note: For examples of how this template is used, see other $RMMC*nn*D members in the CC2DEXEC data set.

Register and Maintain the Macros

To enable an NCL procedure to be used as a macro, register the macro by creating a macro definition for the procedure. Use the Macro Definition panel to define the macro.

Access Macro Definitions

To access macro definitions

1. Enter /**ASADMIN.M** at the prompt.

   The Macro List appears.

Macro Definition Panel

The Macro Definition panel records macro details and identifies the NCL procedure represented by the macro. Depending on the operation you are performing, some or all of the fields on this panel may be modified. Press F1 (Help) for information about the fields.

Add and Maintain Macro Definitions

To add macro definitions

1. From the Macro List, press F4 (Add).

   The Macro Definition panel appears.

2. Complete the following fields:

   Macro

   Specifies the name of the macro.

   NCL Procedure

   Specifies the NCL procedure that this macro represents.

   Press F3 (File).

   The macro definition is saved.

Note: You can subsequently update a macro definition so that it points to a different NCL procedure. The NCL Procedure field is the only field on the panel that can be updated.

Chapter 13: Defining Automation Services Commands

Note: This chapter applies to all products except CA SOLVE:FTS.

This section contains the following topics:

Automation Services Commands (see page 171)
Register and Maintain Commands (see page 171)
Define Your Own Commands (see page 175)

Automation Services Commands

A command is an NCL procedure that performs specific processing. Automation Services comes with a predefined set of commands.

Commands can be generic (that is, perform processing that is not related to a specific resource or service) or can apply to a class, to a type within a class, or to a specific resource or service. A more specific command has precedence over a less specific command.

You can define your own commands to perform site-specific processing by writing an NCL procedure and associating it with a command definition. Supplied command definitions can be modified to suit your requirements.

To execute commands programmatically from a batch process, exit procedure, console, or external application, use the $RMCALL API procedure.

Register and Maintain Commands

Automation Services does not recognize commands unless they are registered by using the command definition facility.

Use this facility to register your own command definitions and to maintain command definitions supplied with Automation Services.

Access Command Definitions

To access command definitions

1. Enter /**ASADMIN.C** at the prompt.

   The Command List appears.

   This panel lists all of the commands that are currently defined, the type of resource or service to which each command applies, and a description of the command. Special classes include ALL, ALL-X, INC, and NONE. For information about these classes, press F1 (Help).

Command Details

The Command Details panel specifies the following:

- The name and description of the command
- Whether the command applies to a class, a type in a class, or a specific resource or service

  Note: For duplicate commands, the region applies the command according to their scope. For example, if you enter A beside an SNA resource, the A command with the class scope of SNA takes precedence over the A command with the class scope of ALL-X.
- The user ID under which the command executes
- Whether the command is executed on remote or local region
- The name of the NCL procedure that performs the processing associated with the command, and the names of any parameters passed to this procedure
- The URL or Java class invoked to perform the processing associated with the command in WebCenter, and the parameters required. To specify a URL, the line must start with URL:. To specify a Java class, the line must start with Java.

Example

The following is the Command Details panel in Browse mode. For more information about the panel, press F1 (Help).

```
PROD1-------------- Automation Services : Command Details -----------$$CMD-0000
Command ===>                                                    Function=BROWSE

- Command Identification and Scope -----------------------------------------
|                                                                            |
| Name ... A          Description ... Activate a Resource                    |
|                                                                            |
| Class ...+ ALL      Type ..........              Name ..                    |
| Security   ANO      Short List..+ ALL                                      |
------------------------------------------------------------------------------
- Execution Details ----------------------------------------------------------|
|                                                                            |
| Command Userid ..+ *      Execute on Remote? ... YES (YES or NO)           |
|                                                                            |
| NCL Procedure .... $RMCONS LCMD=A SYSNAME=&ZRMDBSYSNAME                    |
|                                   VERSION=&ZRMDBVERSION                    |
|                                   CLASS=&ZRMDBCLASS                        |
|                                   NAME=&ZRMDBNAME                          |
|                                                                            |
| Web Action ....... JAVA:com.ca.syd.statusmonitor.gui.StatusMonitorConfirmCo |
|                    mmandHandler                                            |
------------------------------------------------------------------------------
```

Command Prompting, Confirmation, and Validation

The Command Prompting, Confirmation, and Validation panel is used to specify the following:

- Whether the user is prompted, the type of prompting, and if parameters required by the command are not supplied.
- Whether confirmation is required before the command is executed and the name of the (optional) confirmation exit procedure that performs the confirmation.
- The name of the (optional) validation exit procedure that performs validation on command parameters.

Example

The following is the Command Prompting, Confirmation and Validation panel in Browse mode. For information about the field definitions, press F1 (Help).

```
-------- Automation Services : Prompting, Confirmation and Validation ---------
Command ===>                                                 Function=BROWSE

. Prompting ------------------------------------------------------------------.
|                                                                              |
| Type of Prompting Required ... RESOURCE  (SYSTEM, RESOURCE or NONE)          |
|                                                                              |
| SYSTEM   prompts for System Name and Version if not provided                 |
| RESOURCE prompts for System Name, Version, Class and Name if not provided    |
| NONE     defaults to the local System Name and Version if not provided       |
'------------------------------------------------------------------------------'
. Confirmation ---------------------------------------------------------------.
|                                                                              |
| Does This Command Require Confirmation? ... YES  (YES or NO)                 |
| Confirmation Exit ...                                                        |
|                                                                              |
'------------------------------------------------------------------------------'
. Validation -----------------------------------------------------------------.
|                                                                              |
| Validation Exit .....                                                        |
|                                                                              |
'------------------------------------------------------------------------------'
 F1=Help      F2=Split     F3=Exit      F4=Edit
 F7=Backward              F9=Swap                 F11=Panels
```

Add and Maintain Command Definitions

To add command definitions

1. Enter /**ASADMIN.C** at the prompt.

   The Command List appears.

2. Press F4 (Add)

   The Command Details panel appears.

3. Complete the fields, as required. For more information, press F1 (Help).

Note: If you want to define a line command, the name can only be three characters long.

Command definitions can also be browsed, updated, copied, or deleted from the Command List panel.

Define Your Own Commands

You can define your own commands to perform specialized processing. You need to:

- Write an NCL procedure that performs the processing that you want your command to perform.
- Register the command to Automation Services (for more information, see Registering and Maintaining Commands in this chapter).
- Manually distribute the procedure to linked regions or place it in a shared library, accessible to all systems.

You can register different versions of a command with the same name. Automation Services executes the version that best fits the resource or service the command is applied to.

For example, you can register two versions of a command with the name Z, where version one applies to all resources, and version two applies to started tasks only. When the Z command is applied to a started task, Automation Services executes version two of the command.

Code Your NCL Procedure

Your NCL procedure should set RETCODE=0 to indicate successful completion—a non-zero return code indicates a processing failure. You can return a message from the procedure by setting the &SYSMSG variable.

For more information about NCL programming, see the *Network Control Language Programmer Guide* and the *Network Control Language Reference Guide.*

Commands That Execute on Remote Systems

If a command relates to a resource or service on a remote system, you can set the Execute on Remote? field to YES. The NCL procedure associated with the command will then execute on the remote system.

If the command requires a presentation space (that is, the command displays information to the user who issued it), set the Execute on Remote? field to NO. In this case, your NCL procedure must extract data from the remote system and present it on the local system. The name of the system where the data is sourced is stored in the variable ZRMCMDLINK.

Variables Available to a Command NCL Procedure

The following variables can be used in an NCL procedure associated with a command, to pass data required by the command:

ZRMSYSNAME

Contains the name of the system to which the command is to be applied.

ZRMVERSION

Contains the version number of the system to which the command is to be applied.

ZRMCMDUSERID

Contains the user ID of the person who issued the command.

ZRMCMDLINK

Contains the name of a link (which is, in fact, the ACB name of the target region) that will route a command to a region. The ZRMCMDLINK variable contains the ACB name of the current region if the command is not to be routed to a different system.

ZRMCMDPARMS

Contains any command parameters specified by the user. Parameters are in the format parameter=value, separated by spaces. This enables you to override default values.

ZRMDB*

Knowledge base variables (see page 386) have the prefix ZRMDB. If the command is to be applied to a resource or a service defined in the knowledge base, the command procedure needs access to relevant values stored in knowledge base variables. One or more of the following four variables is likely to be required:

ZRMDBSYSNAME

Contains the name of the system image where the resource is defined if the command refers to a specific resource or service.

ZRMDBVERSION

Contains the version number of the system image where the resource is defined if the command refers to a specific resource or service.

ZRMDBCLASS

Contains the class name of the resource or service to which the command is to be applied.

ZRMDBNAME

Contains the name of the resource or service to which the command is to be applied.

ZRMST*

Status variables (see page 390) have the prefix ZRMST. If the command is to be applied to a defined resource or service, the command procedure needs access to relevant values stored in status table variables.

ZMSG*

Message variables (see page 393) have the prefix ZMSG. If the command is to be executed as a result of the receipt of a particular message, the command procedure needs access to relevant values stored in message variables.

Chapter 14: Customizing the Display Attribute Tables

Note: This chapter applies to all products except CA SOLVE:FTS.

This section contains the following topics:

Display Attribute Tables (see page 179)
Edit a Display Attribute Table (see page 180)
Logical State Attributes Table (see page 181)
Automated Mode Attributes Table (see page 182)
Manual Mode Attributes Table (see page 183)
SNA Resource Logical State Normalization Table (see page 185)

Display Attribute Tables

Display attribute tables list the following:

- The range of possible logical states for resources and services
- The display attributes that are used by the status and graphical monitors to show the logical state of resources and services

Edit a Display Attribute Table

To edit and customize the information in these tables

1. Enter **/ASADMIN** at the prompt.

   The Administration Menu appears.

2. Select option **A** - Display Attribute Tables.

   The Panel Display List appears.

3. Select the table that you want to edit.

   The table appears.

4. Press F4 (Edit) and edit the attributes.

5. Press F3 (Exit).

   The changes are saved.

Important! When you make a change to a display attribute table, this change is not reflected immediately. Changes are applied to individual resources and services when an event arrives that forces an update of the resource or service status. This can result in a monitor displaying a mixture of old and new attributes. To avoid this situation, issue the CHECKALL command from the resource or service monitor immediately after making changes to the display attributes tables. This forces the generation of events for all resources and services, and all status displays are updated.

Logical State Attributes Table

The Logical State Attributes Table displays the attributes used by the status and graphical monitors to show the logical states of resources and services. The logical state of a resource or service is derived from the values defined in the Automated and Manual Mode display attribute tables.

The display attributes used in this table are as follows:

- Rank in severity
- Intensity
- Color
- Highlighting

Example

This is a sample logical state attributes table:

```
PROD-------- Automation Services : Logical State Attributes Table --$$ATTR-0000
Command ===>                                                    Function=BROWSE


          .-----------+------+-----------+-----------+-----------.
          | STATE     | RANK | INTENSITY | COLOR     | HIGHLIGHT |
          +-----------+------+-----------+-----------+-----------+
          | UNKNOWN   | 001  | LOW       | WHITE     | NONE      |
          | INERROR   | 002  | HIGH      | WHITE     | REVERSE   |
          | FAILED    | 003  | LOW       | RED       | NONE      |
          | ATTENTION | 004  | LOW       | PINK      | NONE      |
          | DEGRADED  | 005  | LOW       | YELLOW    | NONE      |
          | PENDING   | 006  | LOW       | TURQUOISE | NONE      |
          | STARTING  | 007  | LOW       | BLUE      | NONE      |
          | STOPPING  | 008  | LOW       | BLUE      | NONE      |
          | OK        | 009  | LOW       | GREEN     | NONE      |
          '-----------+------+-----------+-----------+-----------'
```

If you want resources with a logical state of INERROR to be displayed in red and reverse video, at high intensity, you would assign the following values:

Intensity: HIGH

Color: RED

Highlight: REVERSE

Note: Your system uses the blinking highlight attribute to display a resource when an automated action is being performed on the resource; therefore, we recommend that you do *not* use the BLINK highlight attribute.

Logical State Attributes and SNA Network Summary Display

Note: This section only applies to CA NetMaster NA.

Color coding of the summary display also corresponds to the logical state attributes table. ACTIVE and INACTIVE, which do not appear in the table, assume the color of the OK state.

Automated Mode Attributes Table

Note: This section only applies to CA NetMaster FTM, CA NetMaster NA, CA SOLVE:Operations Automation, and CA SOLVE:Operations Automation for CICS.

The Automated Mode Attributes Table lists the logical states assigned to resources and services in Automated mode. The logical state is based on a combination of the mode, the desired state, and the actual state. The valid logical states are listed in the Logical State Attributes Table.

Set the Logical States for the Automated Mode

Set the logical states to best describe the situation. For example, if the actual state of a resource is INACTIVE and the specified desired state is ACTIVE, then in the automated mode of operation, Automation Services will try to bring the actual state to ACTIVE also. Use the PENDING logical state to indicate that the resource is in the process of becoming active.

Using the settings in the Logical State Attributes Table panel shown previously in this chapter as an example, a resource that is PENDING is displayed in low intensity turquoise.

Example

This is a sample Automated Mode Attributes Table.

```
PROD-------- Automation Services : Automated Mode Attributes Table -$$ATTR-0000
Command ===>                                                    Function=BROWSE

                 .-----------+--------------------------.
                 |           |      DESIRED STATE       |
                 | ACTUAL    +-------------+------------+
                 | STATE     | ACTIVE      | INACTIVE   |
                 +-----------+-------------+------------+
                 | ACTIVE    | OK          | PENDING    |
                 | STARTING  | OK          | INERROR    |
                 | STOPPING  | INERROR     | OK         |
                 | DEGRADED  | DEGRADED    | OK         |
                 | INACTIVE  | PENDING     | OK         |
                 | FAILED    | FAILED      | INERROR    |
                 | UNKNOWN   | UNKNOWN     | UNKNOWN    |
                 '-----------+-------------+------------'
```

Manual Mode Attributes Table

The Manual Mode Attributes Table lists the logical states assigned to resources and services in manual mode. The logical state is based on a combination of the mode, the desired state, and the actual state. The valid logical states are listed in the Logical State Attributes Table.

Set the Logical States for the Manual Mode

Set the logical states to best describe the situation. For example, if the actual state of a resource is INACTIVE, and the specified desired state is ACTIVE, then in the manual mode of operation the operator needs to take action. Automation will *not* take place to bring the actual state of the resource to ACTIVE (to satisfy the desired state) without operator intervention. Therefore, use the ATTENTION logical state to indicate to the operator that action is required.

Using the settings in the Logical State Attributes Table panel shown previously in this chapter as an example, a resource that is in a state of ATTENTION is displayed in low intensity pink.

Example

This is a sample Manual Mode Attributes Table.

```
PROD--------- Automation Services : Manual Mode Attributes Table ---$$ATTR-0000
Command ===>                                                    Function=BROWSE


                  +-----------+---------------------------+
                  ¡           ¡      DESIRED STATE        ¡
                  ¡ ACTUAL    +-------------+-------------+
                  ¡ STATE     ¡ ACTIVE      ¡ INACTIVE    ¡
                  +-----------+-------------+-------------+
                  ¡ ACTIVE    ¡ OK          ¡ ATTENTION   ¡
                  ¡ STARTING  ¡ OK          ¡ ATTENTION   ¡
                  ¡ STOPPING  ¡ ATTENTION   ¡ OK          ¡
                  ¡ DEGRADED  ¡ ATTENTION   ¡ ATTENTION   ¡
                  ¡ INACTIVE  ¡ ATTENTION   ¡ OK          ¡
                  ¡ FAILED    ¡ ATTENTION   ¡ ATTENTION   ¡
                  ¡ UNKNOWN   ¡ UNKNOWN     ¡ UNKNOWN     ¡
                  +-----------+-------------+-------------+




 F1=Help      F2=Split     F3=Exit      F4=Edit
 F7=Backward  F8=Forward   F9=Swap                F11=Panels
```

SNA Resource Logical State Normalization Table

Note: This section only applies to CA NetMaster NA.

The SNA Resource Logical State Normalization Table lists the logical states assigned to individual SNA resources. The logical state is based on a combination of the desired state and the actual state. The logical state for each SNA resource can be either OK or NOTOK.

This state normalization table is only used by SNA groups and only when the SNA group has a threshold specified for calculating the logical state of the SNA group. The table lets you classify the resources in a group into two logical states: OK and NOTOK, and the threshold specified is then used to assign the logical state of the SNA group.

Note: This table does not affect the normal operation of SNA resources that are not in an SNA group.

Set the Logical States

Set the logical states to best describe the situation. For example, if the actual state of a resource is INACTIVE, and the specified desired state is ACTIVE, then the logical state of the SNA resource is NOTOK.

Example

This is a sample SNA Resource Logical State Normalization Table.

```
--- Automation Services : SNA Resource Logical State Normalization Table ------
Command ===>                                                    Function=BROWSE


                    .-----------+---------------------.
                    |   SNA     |    SNA RESOURCE     |
                    | RESOURCE  |    DESIRED STATE    |
                    |  ACTUAL   +----------+----------+
                    |  STATE    | ACTIVE   | INACTIVE |
                    +-----------+----------+----------+
                    | ACTIVE    | OK       | NOTOK    |
                    | STARTING  | OK       | NOTOK    |
                    | STOPPING  | NOTOK    | OK       |
                    | DEGRADED  | OK       | NOTOK    |
                    | INACTIVE  | NOTOK    | OK       |
                    | FAILED    | NOTOK    | NOTOK    |
                    | UNKNOWN   | NOTOK    | NOTOK    |
                    '-----------+----------+----------'



 F1=Help      F2=Split     F3=Exit      F4=Edit
 F7=Backward               F9=Swap                 F11=Panels
```

Chapter 15: Customizing the Status Monitor Display Format

Note: This chapter applies to all products except CA SOLVE:FTS.

This section contains the following topics:

Status Monitor Display Formats (see page 187)
Create a Status Monitor Display Format (see page 189)

Status Monitor Display Formats

A status monitor display format determines what information is displayed for managed services or resources on the status monitor.

Default formats are supplied with your product. You can modify the default formats or set up other formats to suit your requirements.

Users can select a format by issuing the FORMAT command when viewing the status monitor. They can also update their user profile to set a default format.

Status Monitor Views

The status monitor technology is used by different monitors for different products. Product-specific monitors are known as views. The following table shows supported views.

View Name	Monitor	Support multi-column displays?
CICS	CICS Resource Monitor	Yes
FILETRAN	File Transfer Resource Monitor	No
FTMON	Active File Transfer Monitor	No
IPNODE	IP Node Monitor	No
IPRSC	IP Resource Monitor	No
NCPVIEW	NCP Monitor	No
PRINTER	Printer Monitor	Yes
RESOURCE	Resource Monitor	Yes
SERVICE	Service Monitor	Yes
TAPE	Tape Drive Monitor	Yes

Multi-column Displays

Some of the views support multi-column displays (up to five columns). The views that support multi-column displays are shown in the table.

In the 1-column display, each line shows one managed item. In the 2-column display, each line shows two managed items. In the maximum 5-column display, each line shows the status of five managed items.

The information shown in each display column is determined by the applied status monitor display format. Default formats are supplied for each of the column displays, but you can define your own formats.

Create a Status Monitor Display Format

To define status monitor display formats

1. Enter /**ASADMIN.L** at the prompt.

   The List Definition List appears.

2. Press F4 (Add).

   The List Description panel appears.

   Note: You can also use the C action code to open a copy of an existing display format definition that you can modify.

3. Complete the following fields:

   Appl ID

   Specifies the ID of the application to which the list belongs. Ensure that the value is $RM.

   List Type

   Specifies the type of list. Ensure that the value is PRIVATE.

   Userid

   Specifies the name of the view, or if you are building a format for a view that supports multi-column displays, then append a number (from 1 to 5) to COLUMNS to identify the column display for which the format is created (for example, COLUMNS1 for a 1-column display). (COLUMNS*n* is the user that owns the format.)

   List Name

   Specifies the name of the format.

   Description

   Describes the format.

   Service Procedure

   Specifies the name of an NCL procedure to execute when displaying a list using this definition. Enter NONE.

   Entry Msg Position

   Specifies the column in which extended display starts.

Entry Msg Length

Specifies the width of the extended display as follows:

- If you want the extended display to be of a specific width, specify the width in columns.
- If you want to allow the width to extend to the end of the screen, leave the field blank.
- If you do not want to show the extended display, specify 0.

Note: Do not change the value of the other fields.

Press F8 (Forward) three times.

The List Format panel appears. The panel provides a text editor window.

4. By using the text editor, enter column headings and variables to specify the information to display on the status monitor.
5. Press F3 (File).

   The format is created.

Example

Here is an example of the completed panel.

```
PROD--------------------- CAS : List Description -------------------Page 1 of 4
Command ===>                                                      Function=Add

Appl ID ..............+ $RM
List Type ............. PRIVATE   (PUBLIC or PRIVATE)
Userid ................ COLUMNS1  (Userid if PRIVATE)
List Name ............. ALLFLDS_
Description ........... All status fields monitor format
Title ................. ____________________________________________________
Status ................ ACTIVE__         Group ..............+ ____________
Service Procedure ..... NONE____         Data Source ......... ____________
Get All Entries? ...... YES              Exit Name ........... ________

Add Allowed? .......... YES              Help Name ........... ____________
Default Mnemonic ...... B__              Select Mnemonic ..... S__
Entry Msg Position .... 33_              Entry Msg Length .... ___
Present Empty List? ... YES              Auto Refresh Rate ... _____
                                         Heading Sub Char .... ___
Comments .............. ____________________________________________________
                        ____________________________________________________
                        ____________________________________________________
                        ____________________________________________________

 F1=Help      F2=Split     F3=File      F4=Save
              F8=Forward   F9=Swap                              F12=Cancel
```

Specify the Status Monitor Display Format

Specify the status monitor display format on the List Format panel.

The following list gives the widths for the five different column displays:

1-column display	75 characters
2-column display	35 characters
3-column display	21 characters
4-column display	15 characters
5-column display	11 characters

Each column contains an extra five positions at the beginning to allow for the command input field. For example, for a 1-column display, the width of the format is 75 characters, representing positions 6 through 80 of the status monitor.

For each type of information you want to display on the status monitor, you need to specify a static heading and a variable that contains the required information.

To specify the status monitor display format

1. From the List Format panel, specify the headings. A heading can be up to 10 lines. These lines are known as heading lines.
2. Specify the corresponding variables beneath the heading lines. Specify the variables in a single line, known as an entry line.

   If the name of a variable is longer than the data to be displayed, create a shorter alias for the name.

Status Monitor Headings

A heading describes the information being displayed under it. Type the headings as you would like them to appear on the status monitor. A heading can contain up to 10 lines of text.

Status Monitor Variables

The variable contains the information you want to display. You can use any of the status variables. For a list of these variables, see the *User Guide*.

In addition, you can use the following special variables:

&ZRMSTNRMCLDS

Contains the:

- Subclass name for USRCLS class resources
- SNA group type for SNAGRP class resources
- Class name for services and other resources

For more information, see the *User Guide*.

&ZRMSTOVRFLAG

Contains the flag that identifies the status overrides applied on a service or resource.

&ZRMSTXNAME

Contains the resource name. If the resource has an accompanying extended display, the name is prefixed with a plus sign (+).

If the display format hides extended displays, this variable enables you to identify the existence of an extended display, which you can view by entering S beside the resource.

Create Shorter Aliases for Variable Names

The name of a variable can sometimes be longer than the displayed data. You can enter a shorter name and then make that shorter name an alias of the actual name.

To create aliases to variable names

1. From the List Format panel, press F5 (Fields).

   The List Entry Line Fields panel appears.

2. The Entry Line Field column contains the variable name you specified in the display format. Type the corresponding real variable name under the Real Field heading.

3. After you have created the aliases, you can perform one of the following actions:

If you want to ...	Press ...
Save the format and exit the format definition panels	F3 (File)
Save the format and remain on the List Entry Line Fields panel	F4 (Save)
Return to the List Format panel	F5 (Format)

Extended Displays

When extended displays are active, they overwrite the last portion of a formatted display screen. The starting position and the width of an extended display are determined by the values in the Entry Msg Position and the Entry Msg Length fields on the List Description panel (page 1 of the format definition panels).

With a multiscreen display format, extended displays affect the first screen only.

Specify a Multiscreen Display Format

You can create a multiscreen status monitor display. A multiscreen display can have up to 10 screens, enabling you to display more information on the status monitor. The screens can be accessed by pressing the F11 (Right) or F10 (Left) function keys from the status monitor.

Example

The following example contains two screens. Lines 0001 and 0002 define the first screen, and lines 0003 and 0004 define the second screen.

```
PROD----------------------- CAS : List Format ---------------------Page 4 of 4
Command ===>                                    Function=Browse Scroll ===> CSR

Appl ID ... $RM      Type.Userid ... PRIVATE.COLUMNS1       Name ... DSPFMT2P

LINE ----+----10---+----20---+----30---+----40---+----50---+----60---+----70---
**** ****************************** TOP OF DATA ******************************
0001 System   Class    Resource     Desired  Actual     Mode       Logical   Ov
0002 &SYSNAME &CLNO    &NAME        &DSTST   &PHYST     &CURMODE   &NRMST    &O
0003 Resource      Current Mode Reason
0004 &NAME         &ZRMSTMODECU0
**** ***************************** BOTTOM OF DATA ****************************
```

Chapter 16: Maintaining Prompt Lists

Note: This chapter applies to all products except CA SOLVE:FTS.

This section contains the following topics:

Prompt Lists (see page 195)
Access Prompt Lists (see page 195)

Prompt Lists

You can enter a question mark (?) in any prompted field to display a *prompt list* (that is, a list of valid values for the field). An authorized user can update these prompt lists if necessary.

Access Prompt Lists

To access the field prompt lists

1. Enter /**ASADMIN.P** at the prompt.

   The Field Prompt List displays descriptions of all prompted fields, in alphabetical order.

Add a Value to a Prompt List

You can add an entry that will be used generically by a specific resource.

To add a value to a prompt list

1. Enter /**ASADMIN.P** at the prompt.

   The Field Prompt List appears.

2. Enter **S** or **V** (FieldValues) beside the Field Prompt Description to which you want to add a value.

   The Field Prompt Entry List appears, displaying the existing values in the list.

3. Press F4 (Add).

   The Field Prompt Entry Definition panel appears.

4. Enter or modify the fields as required. Use variables, a string of less-than characters (<), or a string of greater-than characters (>) to define resource-specific details such as the resource name. You can also use the underline character (_) to define variables.

   Press F4(Save).

   The new value is saved.

Use Variables

Variables in a prompt list entry are substituted by their values when you select the entry for the field. You can disable variable substitution if you want the variable to appear in the field, *not* the value of the variable. Variable substitution can be disabled for variables with the formats &ZRM* and &ZMSG*.

To disable variable substitution, replace the ampersand (&) in front of the variable name with the underline character (_). For example, if you specify _ZRMDBNAME in a prompt list entry and select the entry for the field, &ZRMDBNAME is displayed in the field.

More information:

Knowledge Base Variables (see page 386)
Status Variables (see page 390)
Message Variables (see page 393)

Use Less-than Signs (<) to Represent a Left-justified, Fixed-length Variable Field Value

Some messages contain left-justified, fixed-length fields for resource names. If the name is not the right length, the name is left-justified. To handle left-justified fixed length fields, use less-than signs (<). Each less-than sign represents one character. (You cannot use variables because variables do not provide padding.) For example, <<<<< represents a 5-character field with left justification.

Use Greater-than Signs (>) to Represent a Right-justified, Fixed-length Variable Field Value

Some messages contain right-justified, fixed-length fields for resource names. If the name is not the right length, the name is right-justified. To handle right-justified fixed length fields, use greater-than characters (>). Each greater-than sign represents one character. (You cannot use variables because variables do not provide padding.) For example, >>>>> represents a 5-character field with right justification.

Update Prompt List Definitions

To update a prompt list definition

1. From the Field Prompt List, enter **U** (Update) next to the prompt list that you want to update.

   The Field Prompt Definition panel appears.

2. Complete the following fields:

   Name

   Identifies the field.

   Description

   Describes the prompt list. This description is displayed on the Field Prompt List panel to identify the prompt list. The description is also displayed as the heading on the Field Prompt Entry List panel, which is displayed when you enter a question mark (?) in a prompted field.

   Exclude Display

   Specifies whether this definition is displayed on the Field Prompt List panel.

   Max Abbreviation Length

   Specifies the length of the abbreviated value of an entry in the prompt list. If you do not want to allow abbreviated values, leave the field blank.

   Limits: 3 to 8.

   Max Value Length

   Specifies the length of the full value of an entry in the prompt list.

   Limits: 3 to 52.

   Max Description Length

   Specifies the length of the description of an entry in the prompt list. If you do not want to allow descriptions, leave the field blank.

   Limits: 3 to 38.

Note: Ensure that the sum of the values in the Length fields results in a string that fits into a single line on the display screens. The string starts at column 7 on the display screens.

Chapter 17: Using Linked Product Regions

This section contains the following topics:

Communicate Between Your Product Regions (see page 199)
Link Regions (see page 200)
INMC Links (see page 203)
Control INMC Links (see page 206)
Troubleshooting for INMC Links (see page 210)
Plan INMC Links (see page 211)
Maintain Link Definitions (see page 216)
Provide Centralized Control of Connected Regions (see page 219)
Manage APPC Links (see page 229)

Communicate Between Your Product Regions

Your system services provide the following methods to communicate between regions:

- INMC—Inter-Network Management Connection
- APPC—Advanced Program-to-Program Communication

If two or more regions are connected, you need to be aware of which regions are connected and ensure the links are active when required.

Important! If you use the multisystem operation of Automation Services, all management of INMC links is done internally by Automation Services. In this case, you need to take care if you use the native INMC control as described in this chapter. Be aware that manual actions against INMC and APPC links controlled by Automation Services may disrupt multisystem operation.

Display Links

You can display links from a command entry panel.

To display links

1. Enter the following command at the prompt:

   `SHOW LINKS TYPE=`*`type`*

   type

   Specifies the type of link, that is, INMC or APPC.

Note: You can also display INMC links by entering the shortcut **/INMC**. This method displays more information about the links.

Link Regions

The INMC feature lets you establish and operate links between regions.

The DOMAIN command is used to define a remote region. The command includes the following:

- The name of the region definition. This is the same as the name specified by the PRI JCL parameter.
- The access methods that can be used to contact remote regions
- Access method specific details:
  - For VTAM, the region name is used as the VTAM ACB APPL name.
  - For TCP/IP, a region definition that includes the IP name or the address and port number.

The LINK START command is then used to establish the INMC to the remote region identified by the DOMAIN command.

Inter-Network Management Connection

INMC provides a general-purpose data transport mechanism that allows a region to communicate with one or more regions across a logical link known as an INMC link. An INMC link can be *one* of the following:

- Simple
- Multipath

Simple INMC Links

INMC provides basic functionality, which is suitable for networks where there is only one physical network path between regions. Simple INMC provides the following functionality:

- It allows a region to communicate with one or more other regions across a logical link.
- It allows a maximum of two sessions between any two regions. Each region uses one session as the transmit part of the link and the other as the receive part.
- You can set up simple link definitions, which are stored in the VFS data set, or you can pass all necessary parameters on link start.

Multipath INMC Links

INMC provides extended facilities for complex networks where multiple physical network paths are available between regions. These are known as multipath links.

A multipath link allows a link between two regions to comprise up to sixteen communication sessions (that is, each region opens up to eight sessions with the other region). Traffic is multiplexed across these sessions either equally or with some sessions having preference over others.

With multipath links you set up link definitions, which are stored in the VFS data set.

Note: If you start an INMC link that has no definition, a simple link is started.

Multipath links can provide the following significant advantages over standard INMC capabilities:

- Multiple sessions traverse different network paths.
- Multiple access methods can be used to provide a backup if there are network connectivity problems.
- High-volume INMC traffic (for example, file transmission operations) can be routed across specific network paths.
- The multipath link can use the combined bandwidth of all network paths to achieve high data transfer rates.
- Multipath links can often eliminate the need to provide additional network bandwidth between computer sites, by multiplexing heavy FTS traffic across existing network paths.
- Multipath links provide improved reliability, because the link remains active as long as one connection exists.

Access Methods

Regardless of the link type, INMC links support VTAM, TCP/IP, XNF, or EPS access methods.

Note: The VTAM application definition that you use for general communications (that is, the APPL identified by the PRI JCL parameter) should include the PARSESS=YES operand. This parameter is required to support the parallel sessions that make up any INMC links that use VTAM.

Security

INMC provides a security exit, allowing verification of the identity of the remote region.

Note: For more information, see the *Security Guide*.

Traffic Flow

INMC is used by the following system components to transmit and receive data traffic flow:

- Remote Operator Facility (ROF)
- Inter-System Routing (ISR)
- CA SOLVE:FTS

Traffic Flow on Simple Links

Data traffic flows between two domains across the twin-session link between the two regions.

Traffic is directed according to preference, across the session in which the transmitting region is the primary end. Therefore, the link can be regarded as having a transmit part and a receive part for each region. The two sessions operate independently.

If there is only one active session, then the traffic flows across the session in both directions.

Traffic Flow on Multipath Links

You can control the way that traffic is prioritized and the routes that the traffic takes by defining specific types of multipath links (see page 211).

INMC prioritizes the traffic so that ROF messages are scheduled with the highest priority and CA SOLVE:FTS the lowest. The multipath link definitions can be defined to control ROF and ISR traffic; however, CA SOLVE:FTS traffic is controlled as part of the CA SOLVE:FTS initiator definition.

Control Links

Links are established by using the LINK START command.

Link definitions are retrieved from the VFS data set.

Links are stopped by using the LINK STOP command, and can be reset by using the LINK RESET command.

INMC Links

There are two types of INMC links:

- Static INMC links
- Dynamic INMC links

Static INMC Links

Static INMC links are most useful between regions that regularly communicate with each other and that always use the same region names.

The link between two regions cannot be established unless each region has been defined to the other and is prepared to accept the connection.

Dynamic INMC Links

Dynamic INMC links provide a mechanism for establishing transient INMC connections between regions to satisfy temporary connection requirements, without specifying permanent definitions for particular regions.

Dynamic links are established between pairs of regions. They occur when *both* of the following occur:

- One region actively attempts to establish communications.
- The targeted region is prepared to accept the incoming INMC request on a generic naming basis, without having been specifically instructed to communicate with the other region.

Dynamic link capability is controlled by the DEFLINK command. This command defines the parameters necessary for an INMC link request to be accepted from a remote region without issuing any LINK START commands.

The DEFLINK command specifies the following critical parameters:

- A mask value used to decide whether the INMC link request should be accepted
- A prefix value used to generate the link name of the INMC link that will be created if the request for the connection is accepted

Establish INMC Links

Use the LINK command to define regions to each other.

You can enter the LINK command from OCS or include it in the RMREADY procedure. The LINK command can specify the following to the domain:

- The access method to use (VTAM, XNF, EPS, or TCPIP).
- The link name by which the remote domain is known by this region and by the operators of this region. This can be up to 12 characters long and lets you assign meaningful names to the various regions in your network. It might help if you always assign a link name with the same value as the system ID in the SYSTEMID parameter group for the remote region.
- The name of the region definition that contains the access method details.
- A message prefix to add to all messages received from the remote region.
- The color and highlighting that is to apply to all messages received in OCS from the remote region.
- The retry interval at which your system attempts to contact the remote region following a link outage.

Example: Establish INMC Links

To establish a link with a remote region in Chicago with a region name of CHIC1472 and messages from the remote region displayed in red, blinking, and with a prefix of CH14, issue the following commands:

```
DOMAIN DEFINE CHIC1472 VTAM=YES

LINK START=CHICAGO DOMAIN=CHIC1472 COLOR=RED HLIGHT=BLINK MSGID=CH14
```

Note: If you are establishing a static link, to start communication between the two regions, you must also enter a LINK command definition to this region from the CHIC1472 region.

Routing Commands

When the link is established you can route commands to the remote region by using the link name.

Example: Routing Commands

To show the users of a remote region called CHICAGO, use the following command:

```
ROUTE CHICAGO SHOW USERS
```

Communication Recovery

If contact is lost on an INMC link, then the action of the regions depends on whether the link is static or dynamic.

Static Links

If contact is lost on a static link, then both the local and the remote regions attempt to re-establish communication automatically.

Dynamic Links

When a dynamic link is established, it operates in the same way as a static link. However, if contact is lost and the dynamic link is broken, then each region acts differently:

- The region that requested the link perceives it as a static link definition and automatically tries to reconnect.
- The region that accepted the link as a dynamic link takes no action to restore the link. It accepts any future request that meets the requirements of a DEFLINK definition.

Control INMC Links

In addition to defining static INMC links, the LINK command is also used to stop and restart links and to delete entire link definitions from storage. You can enter LINK commands from the OCS panel at any time, or you can include them in the READY procedure to have the region attempt link establishment to other regions automatically.

Start an INMC Link

To start an INMC link

1. Issue the following command:

   `LINK START=`*`linkname`*

   linkname

   Specifies the name by which the remote region is known.

   If no previous LINK START command has been issued for this destination, INMC retrieves the link definition stored under the link name from the VFS database and attempts to open an INMC link to the remote region. The number of sessions to be opened and the manner in which they operate is defined in the link definition.

   If no link definition exists on the VFS database, INMC accepts the LINK START command and defines a static INMC link (accepting any operands if supplied).

   If DOMAIN or APPLID and MSGID operands are included on the LINK START command but a link definition already exists on the VFS database, the extra operands are ignored and the VFS definition used.

Stop an INMC Link

To stop an INMC link

1. Issue the following command:

   `LINK STOP=`*`linkname`*

   linkname

   Specifies the name by which the remote region is known.

   This command terminates all sessions with the remote region identified by the *linkname*, and rejects any attempts by the remote domain to reopen the link. The link remains inactive until a subsequent LINK START command is issued.

Specify a Particular Route for INMC Traffic

This method applies only to links that use the VTAM access method.

To allocate an INMC session to a particular virtual route, a COS (Class of Service) definition is specified in the logmode table entry. A COS definition defines a list of virtual routes, in order of preference.

To enable the use of a COS definition, define the following logmode table entry:

```
MODEENT   LOGMODE=tablename, *
          FMPROF=X'12',      *
          TSPROF=X'04',      *
          PRIPROT=X'F0',     *
          SECPROT=X'F0'      *
          RUSIZE=X'8787',    *(or as required)
          COMPROT=X'0000',   *
          COS=cosname
```

This logmode table entry is especially useful when using CA SOLVE:FTS. It ensures that data transfer does not compete with other traffic.

We recommend that when defining multipath links, you associate each session with a separated logmode table entry, where each logmode entry has a separate COS definition that defines a single virtual route.

Note: For more information about logmode table definitions, see the appropriate VTAM installation guides.

Reset an INMC Link

To reset an INMC link

1. Issue the following command:

   ```
   LINK RESET=linkname
   ```

   linkname

   Specifies the name by which the remote region is known.

Note: This command is valid only if the link has already stopped. It removes the current link definition information from storage. A subsequent LINK START command retrieves the definition again from the VFS database, if it exists. A link definition modified through INMC maintenance services can be brought on line only if the current active link definition is first reset.

Display INMC Links

To display the current status of an active link to a remote region

1. Enter **/INMC** at the prompt.

   The INMC : Link Status List panel appears. This lists the sessions linked to the remote region.

2. To view more details about a link's status, enter **S** or **B** beside the link name.

   The INMC : Link Status Display panel appears.

   The information displayed on this panel varies slightly depending on the following:

   - Whether the link is defined and has a description
   - Whether the link has ever been active

Improve INMC Link Performance

To improve the performance of your INMC link, you can make the following changes:

- Increase the size of the INMC internal transmission buffers.
- Increase the RU size specified in the logmode table definition (VTAM only).
- Route the INMC link through a particular route (VTAM only).

Increase Transmission Buffer Size

If you are using a very high-speed link to carry INMC traffic, for example, channel-to-channel connections or microwave links, then use the SYSPARMS INMCBFSZ operand to increase the size of the principal INMC transmission buffers. The default size is 4 kilobytes; the maximum size is 15 kilobytes.

Example

To increase the size of the transmission buffer to 8 kilobytes, enter the following command:

```
SYSPARMS INMCBFSZ=8
```

Increase RU Size in Mode Table Definitions

Note: This method applies only to links that use the VTAM access method.

The RU size specified on the BIND parameters when INMC sessions are established should match or exceed the size specified by the SYSPARMS INMCBFSZ operand.

If you use SYSPARMS INMCBFSZ to increase the INMC internal buffer size, you must also specify an RU size of at least the size of the INMC buffers for the sessions that are established. If you do not do this, you will not gain any benefit from the increase. If you specify RU sizes greater than the INMC internal buffer size, then the excess is not used.

The RU size is specified by the RUSIZES parameter of the logmode table entry.

The minimum RU size permitted for INMC sessions is 256 bytes. The recommended RU size is 1 KB. If no RU sizes are specified or if a value less than the minimum size is set, INMC uses a default value of 256 bytes.

Use Security with INMC Links

INMC provides a security exit that allows you to implement security to determine whether a link between two regions should be established.

An assembler exit point is provided by INMC to pass control to an installation-coded routine. This routine communicates with an equivalent routine at the remote region and exchanges any identification information that might be required.

For information about the parameter lists and coding requirements for the INMC security exit, see the *Security Guide*.

Diagnose Problems with INMC Connections

If problems occur when establishing INMC sessions between regions, then a tracing facility is provided to help you determine problems. This is provided by setting the SYSPARMS LNKTRACE and SESSMSG operands.

Troubleshooting for INMC Links

For any given pair of regions with defined links, establishing a connection should be automatic once there is a path to each region through the network.

If a link cannot be activated, check the following:

- Has each region been defined to the other? If not, see your systems administrator.
- Does a SHOW LINK display in either region have a link status of PEND-ACT? If so, check that both regions are active, and check the status of the appropriate VTAM cross-region resource managers and resources.
- Does a SHOW LINK display in one region indicate a link status of ACTIVE, but show as STOPPED or PEND-ACT in the other region? If so, this is a system error and you should see your systems administrator. The link definitions for both regions should be stopped, reset, and redefined.
- Does a SHOW LINK display give the status in one region as PEND-ACT, and as STOPPED in the other region? If so, issue a LINK START=*linkname* for the INACTIVE definition.
- If both regions indicate a status of PEND-ACT for the link while there appears to be a network path open between the regions, see your systems administrator.
- Does a SHOW LINK command in either region have a link status of FAILED? If so, check backwards through the activity log for error messages recorded at the time of the failure. Refer these to your systems administrator. The link can be restarted using a LINK START command.

Plan INMC Links

A link between two regions can comprise one or more communication sessions, termed a session group. Each region can establish and operate up to eight sessions with the remote region. The remote region acts as the secondary end of each session.

The simplest link consists of two sessions between two regions, one in each direction. However, the link still functions successfully when only one session is open between the two regions.

The following categories of link can be established between regions:

- Simple-mode links
- Rotate and backup multipath links
- Preferential multipath links

Two regions that are to communicate across an INMC link do not need to have the same category of link in both directions. The traffic flow from each region is controlled according to the link definition.

The choice of category depends upon both your network topography and the network traffic requirements. The following sections describe each type of link in detail so that you can choose the type of links most appropriate for your installation.

Plan Simple-mode Links

A simple-mode link allows each region to maintain a single session with the other domain. All traffic across the link flows along the same physical network path. If the network path fails, the link is broken. This type of link is best suited to network configurations that provide a single path between two regions.

You can define a simple-mode link as follows:

- With default session parameters
- With specific session parameters

Simple-mode Link with Default Session Parameters

You can create a simple-mode link definition by specifying the minimum amount of information, which is:

- The region name of the remote region
- The one- to four-character MSGID used as a prefix on ROF messages received from the remote region

All other session parameters are allowed to default as dictated by the access method when sessions are established.

This way of defining regions to each other is appropriate in the following environments:

- There is a single physical network route providing the communication path between the two regions.
- There is no FTS traffic between the two CPUs.
- There is FTS traffic between the two regions, but its effect on other traffic sharing the network path between the two is of no concern or has been minimized.
- The level of VTAM in use does not support multiple network paths between subareas.

Simple-mode Link with Specific Session Parameters

A simple-mode link with specific session parameters has an associated COS definition for each session so that each session has a particular route or transmission priority.

Note: This applies only to sessions using the VTAM access method.

The COS definition is specified in a logmode table entry for the link definition.

Note: The logmode table entries associated with sessions are always governed by the definition of the remote domain's VTAM APPL.

This method of defining an INMC link is appropriate in the following environments:

- There is only one physical network route connecting two regions and traffic is to be allocated a particular transmission priority.
- For operational reasons INMC traffic is to be restricted to a particular network route that can be enforced through the appropriate COS definition.

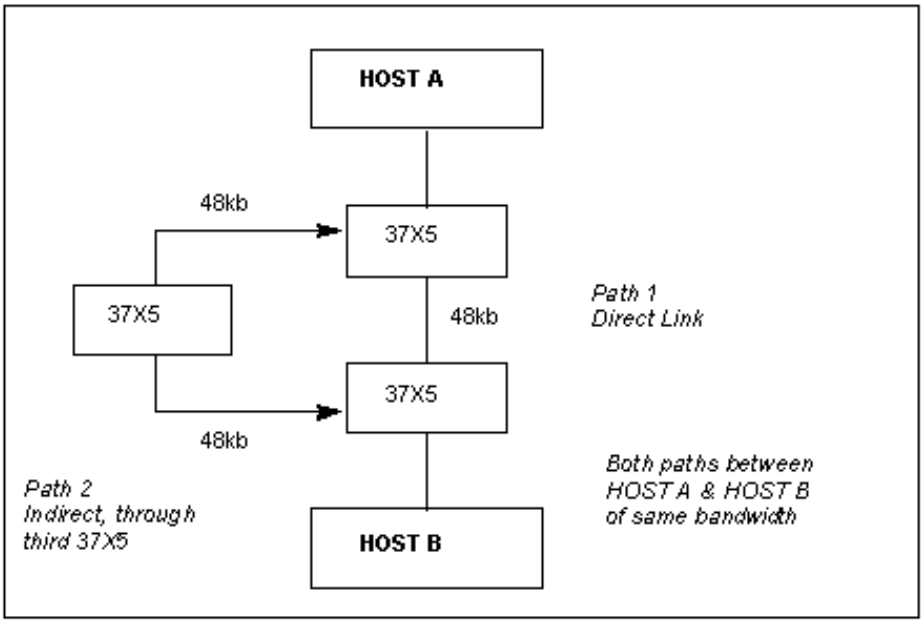
Plan Rotate and Backup Mode Multipath Links

Rotate and backup mode links provide multiple paths through a network connecting two regions. This type of link can establish from two to eight sessions with the remote region. Each session is classified as active or backup (at least one session has to be defined as active). Individual sessions are allocated to different network routes through the use of the appropriate COS definitions.

Traffic between the two regions is sent across all the active links in rotation, and therefore distributed equally across all the active sessions in the session group. If one of the active sessions is lost, then the first available backup session takes its place. When the active session is restored, the backup session is placed back in reserve.

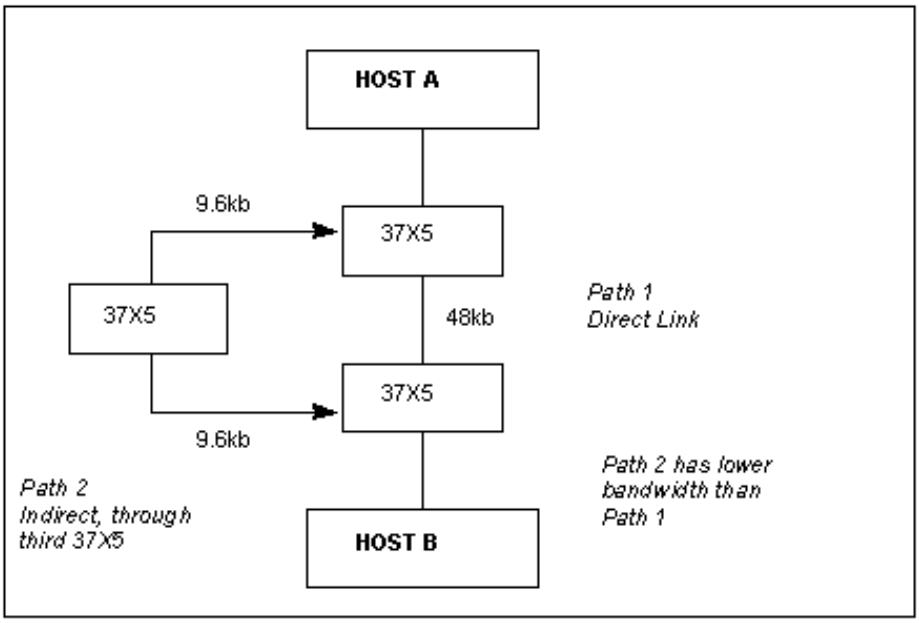
Example 1: Similar Bandwidth Paths

Rotate and backup links are appropriate in a network where there are two or more active physical network paths of similar bandwidth between two regions. A link is required that uses the aggregate bandwidth of all those physical paths, as shown in the following example.



Example 2: Different Bandwidth Paths

Rotate and backup links are appropriate in a network where there are two active physical paths of different bandwidth between the two regions. A link is required that uses the high-speed path exclusively, but switches automatically to the slower path if the fast path fails. The following example shows such a configuration.



Rotate and backup links are not suitable for environments where the active sessions traverse different routes with widely differing transmission speeds. This should be taken into consideration when planning the COS definitions to be associated with the various sessions that form the INMC link.

It is recommended that each session be allowed to travel only on one virtual route. This ensures that if a session is active its network path is always the same.

Plan Preferential Mode Multipath Links

A preferential link definition defines from two to eight sessions for a remote region, where each session is allocated a traffic weighting. Data is sent, by preference, on the first session. If the first session is busy, then data is sent on the second session unless that is busy too, and so on.

You can select the traffic level at which a session becomes busy and causes an overflow to the next session. You can also specify the number of unresponded chains of data that can be outstanding on a given session at any one time.

A preferential link is best suited to the following circumstances:

- You want to direct as much traffic as possible across a particular network path to reduce its impact on other network traffic flowing on other network paths. Overflow onto a second network path is allowed only if the volume of traffic exceeds a certain threshold.
- When traffic can be directed to lightly loaded network routes such as those used as backup links
- When additional bandwidth can be borrowed from time to time from other network paths if the preferred path becomes congested

Traffic Synchronization on Rotate and Backup and Preferential Multipath Links

When traffic is distributed across the various sessions that make up a rotate and backup or a preferential link, it is resequenced to ensure that it is presented to the remote regions in the same order as it left the transmitting region. This occurs regardless of network route bandwidths which could lead to faster transmission across one session than another.

Failure of one session in a multi-session link does not disrupt the logical link. Traffic traversing the session at the time of failure is resequenced as required and retransmitted across one of the surviving sessions.

Maintain Link Definitions

Before you can establish multipath links, you must specify a link definition in each region that is going to communicate with another.

Note: You can also define simple mode links. This lets you specify session parameters such as the logmode table entry name.

These link definitions are defined by using the INMC link definition facility. You can define the following types of link:

- Simple-mode link
- Rotate and backup
- Preferential

The link name is the symbolic name by which the remote region is identified. You should choose a link name that is meaningful, so that specific regions can be readily identified by their link name. It is also recommended that each region is known to all other regions by the same link name.

The link definitions are saved on the VFS database for retrieval when a link to a remote region is to be activated.

Only authorized users can define INMC link definitions.

Display INMC Link Definitions

To display a list of INMC link definitions

1. Enter **/INMCDEF** at the prompt.

   The INMC : Link Definition List panel appears, showing an alphabetic listing of all INMC link definitions on the VFS database.

   For a description of the fields shown on this panel, press F1 (Help).

Create an INMC Link Definition

The procedure for creating an INMC link definition varies, depending on the type of link:

- Simple-mode link
- Rotate and backup
- Preferential

These procedures are described in the following sections.

Create a Simple-mode Link Definition

To create a simple-mode link definition

1. From the INMC : Link Definition List, press F4 (Add).

   The INMC : Link Definition Details panel appears.

2. Complete the following fields:

   Destination Link Name

   Specifies the link name that you want to add.

   Limits: 1 to 12 characters

   Transmission Mode

   Specifies the mode of transmission. Enter **S** (Simple).

   Press F3 (File).

   The definition is saved on the VFS database. The INMC : Link Definition List panel appears, with a message confirming successful addition of the definition.

Create a Rotate and Backup INMC Link Definition

To create a rotate and backup INMC link definition

1. From the INMC : Link Definition List, press F4 (Add).

   The INMC : Link Definition Details panel appears.

2. Complete the following fields:

   Destination Link Name

   Specifies the link name that you want to add.

   Limits: 1 to 12 characters

   Transmission Mode

   Specifies the mode of transmission. Enter R (Rotate).

   Press F5 (Sessions).

   The INMC : Rotate and Backup Session Details panel appears.

   This panel lets you specify details of the number of sessions to open, the logmode table entries associated with each of those sessions, and the session. For a description of the fields, press F1 (Help).

3. Complete the fields for each session required and press F3 (File).

   Note: You must define at least two sessions.

   The definition is saved on the VFS database. The INMC : Link Definition List panel appears, accompanied by a message confirming successful addition of the definition.

Create a Preferential INMC Link Definition

To create a preferential INMC link definition

1. From the INMC : Link Definition List, press F4 (Add).

   The INMC : Link Definition Details panel appears.

2. Complete the following fields:

   Destination Link Name

   Specifies the link name that you want to add.

   Limits: 1 to 12 characters

   Transmission Mode

   Specifies the mode of transmission. Enter P(Preferential).

   Press F5 (Sessions).

   The INMC : Preferential Session Details panel appears.

   You can specify details of the number of sessions to open, the logmode table entries associated with each of those sessions, and the traffic thresholds where traffic is distributed onto successive sessions. For a description of the fields, press F1 (Help).

3. Complete the fields for each session required, and press F3 (File).

   Note: You must define at least two sessions.

   The definition is saved on the VFS database. The INMC : Link Definition List panel appears, accompanied by a message confirming successful addition of the definition.

Modify an INMC Link Definition

After an INMC link definition is created, it can be updated at any time.

To modify any of the values specified in a link definition

1. From the INMC : Link Definition List, enter **U** beside the name of the link that you want to update.

   The same panels that were used to add a new definition appear with the values that were originally entered.

2. Enter the required values on the panels and press F3 (File).

   The INMC : Link Definition List panel appears, accompanied by a message confirming successful update of the definition.

Delete an INMC Link Definition

To delete a link definition

1. From the INMC : Link Definition List panel, enter **D** beside the name of the link that you want to delete.
2. Press Enter again to confirm the deletion.

   The link definition is deleted.

Provide Centralized Control of Connected Regions

The following features provide services for centralized control of regions connected using INMC:

Remote Operator Facility (ROF)

Provides centralized control at the *user* level

Inter-System Routing (ISR)

Provides centralized control at the *system* level

You can use the Remote Operator Facility (ROF) for centralized control of regions connected using INMC.

Use ROF for Centralized Control of Remote Regions

ROF lets you monitor and control remote regions from the local region through OCS. ROF uses INMC as its transport facility. ROF services are available only between two regions when there is an active INMC link between them.

Define User IDs for ROF

When defining a user ID so that it can be used to operate ROF sessions, the following questions should be considered:

- What authority level does the user need to issue the commands required to control remote regions?
- What NPF message and resource partitioning is to apply to the user?
- What types of messages are required for them to receive, and how are these messages to be delivered?
- Is the user ID defined on all the remote regions to which it can log on?
- Do the regions have a separate or shared UAMS data set?

Display ROF Users

To identify which users are operating ROF sessions, either to or from remote regions, use the SHOW USERS command.

Include Simplified Command Definitions in INIT and READY

The EQUATE command is used to simplify long and complicated commands that ROF operators have to issue frequently. You can use the EQUATES parameter group (enter **/PARMS**) to make EQUATE commands available as system-wide defaults.

Use ROF with NCL Processes

Any NCL procedure started by a ROF operator can be used to issue commands to remote regions. The results of these commands are returned to the OCS window of the ROF operator who started the procedure.

&INTCMD can also be used from in an NCL procedure to issue commands to remote regions. The results from these commands are returned to the response queue of the issuing procedure and can be read using &INTREAD.

Use ROF from Background Environments

Background environments can create ROF sessions with remote regions. When a ROF session is started, a standard ROF signon occurs using the background environment's user ID. If this user ID is not defined to the remote region then the signon fails.

Background environments can route commands to a remote region over a ROF session and have the results returned to it for examination.

Use ROF from the System Console

System consoles can establish ROF sessions with remote regions and issue commands to that domain. If your console user ID has a UAMS definition, then the user ID must be defined on the remote region if a ROF session is to be established.

If you are using the default console user ID (.DFLTOP), then ROF sessions are not supported.

Sign on to a Remote Region

To sign on to a remote region, use the SIGNON command.

Example

To sign on to the remote region PROD01, enter the following command:

```
SIGNON PROD01
```

The remote region checks to see if you are authorized and then notifies you of the connection.

When the connection is established, you receive any unsolicited messages from that region.

Sign on to a Remote Region over an Inactive Link

If there is no active INMC link to the remote region you want to sign on to, you can issue a SIGNON command to wait in a queue for the INMC link to become active.

Example

To sign on to PROD01 but the INMC link to it is not active, enter the following command:

```
SIGNON PROD01
```

This places your signon in a queue and waits until the INMC link to the remote region becomes active. Once the link becomes active, the ROF session is established and you are notified of this by the system.

All ROF sessions and queued signons are canceled when you exit OCS.

Restart a ROF Session

If an INMC link to a remote region fails while you are connected, your ROF sessions to that region are terminated and queued for reestablishment when the link becomes active again. You are notified when the ROF session is reestablished.

If you exit OCS before the link becomes active, the queued SIGNON is canceled.

Issue Commands on a Remote Region

To issue a command on the remote region, use the ROUTE command.

Example

To send a SHOW USERS command to PROD01, enter the following command:

```
ROUTE PROD01 SHOW USERS
```

The SHOW USERS command executes in PROD01 as if it had been entered from a local terminal, and the result of the command is returned to your OCS window.

Note: If you route a command to a remote region that you are not signed on to, a signon to that remote region is automatically initiated.

Use Command Separators

The ROUTE command allows support for a single embedded colon (:) character as a command separator. When the ROUTE command processor encounters a single colon in the command string, it substitutes a semicolon. If the processor encounters two colons, it eliminates the first and sends the second to the target region as part of the transmitted command.

Simplify Remote Command Execution

Sending commands to remote regions for execution can be simplified by using EQUATE command strings. For more information about the EQUATE command, see the online help.

Issue Commands from a Remote Region

You can issue a command on another remote region through your current ROF session. This lets you establish ROF sessions to remote regions through any number of intermediate regions.

Example

To send a SHOW USERS command to PROD02 through PROD01, enter the following command:

```
ROUTE PROD01 ROUTE PROD02 SHOW USERS
```

This command sends the SHOW USERS command to PROD02 through an intermediate ROF session with PROD01. This provides an alternative to a direct ROF session, and can be useful if direct contact cannot be established with the target region.

Receive ROF Messages

Whenever you receive messages from remote regions, whether as unsolicited information or in response to a command, each message is color coded and prefixed with a message ID.

Use Color Coded Messages

The messages from remote regions can have different colors depending on which remote region they originate from. This helps you to identify the source of a message if you are connected to multiple remote regions.

The SIGNON command lets you override the default color/highlight attributes which were specified when the INMC link to the remote region was established.

Example

To specify the color red for any messages received from PROD01, enter the following command:

```
SIGNON PROD01 COLOR=RED
```

Use the Message ID

By default, ROF messages returned from a remote region across an ROF session are prefixed with a 1- to 4-character identifier of the remote region specified when the INMC link was established.

The SIGNON command lets you override the ROF message ID prefix default. This prefix can be any 1- to 8-character string.

Example

To specify the message ID for any messages received from PROD01 as TEST, enter the following command when you sign on:

```
SIGNON PROD01 ID=TEST
```

Your prefix, and not the default PROD01, appears as the first word of each ROF message from the remote region.

Such prefixes are private to you and can be varied for each of your OCS windows, or be different for each SIGNON command issued by NCL processes executing in your NCL processing region.

Sign Off a Remote Region

To sign off a remote region, use the SIGNOFF command.

Example

To sign off PROD01, enter the following command:

```
SIGNOFF PROD01
```

This command is used for explicit signoff; however, if you exit OCS, all ROF sessions terminate automatically.

Use ISR for Centralized Control of Remote Domains

Inter System Routing (ISR) is used to send data from one region to other regions over an INMC link. This provides the following functionality:

- Centralized control—receive all system console messages, unsolicited messages, network messages, and all associated network status and error information in a central location
- Distributed processing—send specific information to certain regions

The following system environments can use ISR to forward the data they receive to other regions:

AOMPROC

Receives system console messages and related information.

PPOPROC

Receives unsolicited VTAM messages.

CNMPROC

Receives CNM data from VTAM.

CA NetMaster NM for SNA

Receives network information concerning sessions and resources.

To ISR, each of these environments is a conversation class. When they use ISR to send data, they establish a conversation with the environment of the same class in the remote region.

ISR Planning

Before activating ISR connections throughout the network, you should establish a plan for processing data in the ISR capable environments. The following factors should be considered:

- The rate at which messages are received in remote regions
- In which regions you require ISR to run
- The types of messages you want to receive. For example, you might want to filter out messages of a certain severity.

Message Loop Protection

When data exchange takes place using ISR, the following rules apply:

- ISR does not return data to the region from which it was sourced.
- ISR does not return data to the region from which it last came.

This protection is applied to avoid the possibility of one or more messages being continually sent from region to region in a message loop.

However, in some configurations, a message, sourced from outside a group of connected regions, could bypass the above rules and enter a message loop. You should always be aware of this when planning your ISR connections. Some conversations provide additional protection against such an occurrence.

Establish ISR Conversations

Note: This section applies to only CA NetMaster NM for SNA and CA NetMaster NA.

To establish an ISR conversation, use the ISRIN and ISROUT parameter groups. These parameter groups let you enable, disable, or alter a conversation. They also let you control the type of data that is transferred across an INMC link. Each conversation class uses its own version of the ISR command.

To define ISR inbound parameters

1. Enter **/PARMS** at the prompt.

   The Customizer : Parameter Groups panel appears.

2. Enter **U** beside the ISRIN parameter group in the Interfaces category.

   The ISRIN - ISR (Inbound) panel appears.

3. Specify link names for PPO messages, CNM data, and SAW data.
4. Specify whether these links are to NetView (PPO messages and CNM data only.
5. Press F6 (Action).
6. Press F3 (File).

To define ISR outbound parameters

1. Enter **/PARMS** at the prompt.

   The Customizer : Parameter Groups panel appears.

2. Enter **U** beside the ISROUT parameter group.

   The ISROUT - ISR (Outbound) panel appears.

3. Specify link names for PPO messages, CNM data, and SAW data.
4. Press F6 (Action).
5. Press F3 (File).

Monitor the Status of ISR

You can use the SHOW ISR command to display the status of all ISR conversations. This command provides information about the conversation classes that are requested to be enabled, and the type of data being exchanged.

The following types of status are displayed:

Local status

Displays information about the ISR conversations from the perspective of the local region.

Remote status

Displays information about ISR conversations from the perspective of the remote region.

Actual status

Displays the status of the ISR conversations from the combined perspective of both the local and the remote region

Example

The following is an example of SHOW ISR command output:

```
(11.42)---------- NetMaster Operator Console Services (PROD)------------------
  show isr
  N73610 LINKNAME     DMN      STATUS   SSCP     NETID    L-C R-C A-C   QMAX
  N73611               CLASS    -LOCAL  STATUS- -REMOTE STATUS- -ACTUAL STATUS-
  N73612                        E/D SOL IN  OUT E/D SOL IN  OUT E/D SOL IN  OUT
  N73615 *DEFAULT*    -        -        -        -        *   -   -     320K
  N73617               PPO      DIS YES NO  NO  -   -   -   -   -   -   -   -
  N73617               NTS      DIS YES NO  NO  -   -   -   -   -   -   -   -
  N73617               NEWS     DIS YES NO  NO  -   -   -   -   -   -   -   -
  N73617               AOM      DIS YES NO  NO  -   -   -   -   -   -   -   -
  N73615 PROD3        ST3N     ACTIVE   SDD1VTM1 FTI      NO  YES NO    320K
  N73617               PPO      DIS YES NO  NO  ENA YES NO  YES DIS YES NO  NO
  N73617               NTS      ENA YES YES NO  ENA YES NO  YES ENA YES YES NO
  N73617               NEWS     ENA YES YES NO  ENA YES NO  YES ENA YES YES NO
  N73617               AOM      DIS YES NO  NO  ENA YES NO  NO  DIS YES NO  NO
  N13503 *END*
_______________________________________________________________________________




 ==> show isr
```

Control of the Type of Data Received Through ISR

For AOM, PPO, and Network Error Warning System (NEWS) (a component of CA NetMaster NM for SNA), you can control data exchange though use of AOMPROC, PPOPROC, and CNMPROC respectively. For NTS, control is through the region and class parameter settings that request the type of data that NTS is to collect.

Note: For more information about the interaction of NEWS and NTS with ISR, see the CA NetMaster NM for SNA *Administration Guide*.

However, in the case of AOM, PPO, and CA NetMaster NM for SNA, there are important parallels that provide a general understanding of ISR operation in these environments.

The following functionality is common to AOMPROC, PPOPROC, and CNMPROC:

- Each of the environments in which each of the procedures resides has an ISR capability. This capability exists whether or not an actual procedure is running in the environment at any time.
- The ISR command controls the data exchange between the environments. Internal or programmatic control is possible in a procedure by operands on the &*xxx*CONT and &*xxx*DEL NCL statements.

  Note: For the purposes of this topic, *xxx* refers to PPO, AOM, and CNM.
- All the environments have an &*xxx*READ, &*xxx*CONT, &*xxx*DEL, or &*xxx*REPL that can be used to assist with the reading and distribution of the data they receive.

  A procedure reads the data it receives, processes the input, then repeats the process. If no &*xxx*CONT, &*xxx*DEL or &*xxx*REPL statements are issued during processing then by definition an implied &*xxx*CONT is performed.

  Note: For more information about &*xxx*READ, &*xxx*CONT, &*xxx*DEL, and &*xxx*REPL, see the *Network Control Language Reference Guide.*
- Delivery of data to other regions through ISR can be handled as follows:

  Implicit delivery

  No specific destination information is specified for the data.

  For example, if there are no ISR connections enabled, no ISR delivery takes place. However, if there are links enabled for outbound messages, then a copy of the data is sent over every link. Where &*xxx*CONT is used, the NCL return code and feedback information indicate the result of the request.

  Explicit delivery

  Specific destination operands are specified on the &*xxx*CONT and &*xxx*DEL statements. These operands are LINK, DOMAIN, and SSCPNAME.

Note: Only one of these operands can be specified at any one time. Not all components support the use of all these operands. For example, if you want to specify that data received from AOMPROC is only sent to one of your two remote regions, identify this region on the &AOMCONT statement.

Run ISR Data Flow Without a Procedure

If a procedure terminates, or no procedure is started, your system can continue to receive messages sent to the particular environment.

When input arrives for an environment in which no current AOMPROC, CNMPROC, or PPOPROC is executing, it is distributed as though an implicit &*xxx*CONT was issued. Therefore, all messages can be sent to a destination domain for processing by running without a procedure in the source region.

Manage APPC Links

APPC lets you establish links between regions and between a region and other applications. If you have NCL applications that use APPC, you must create APPC links.

Start APPC Links

To start an APPC link, use the LINK command and specify the type of link, with the link name of the region or application you are connecting with.

Example

To establish an APPC link with NMA from SOLVE01, enter the following command:

```
LINK TYPE=APPC START=NMA
```

Sometimes you may have to supply a password before an APPC session is established. This is set up in the APPC link definition by your system administrator.

APPC over an INMC Link

You can start an APPC link that uses INMC as its access method (transport provider).

To start this kind of APPC link

1. Enter the following command:

   ```
   LINK TYPE=APPC START=NMA AM=INMC
   ```

   The link is not established until the INMC link to NMA is active.

Note: Because an INMC link can use TCP/IP as its transport provider, starting an APPC link over an INMC link means that you can establish an APPC link where there is no physical VTAM network connecting the two regions.

Display APPC Link Status

The SHOW LINK command is used to display APPC link status.

Example

To find out the status of all APPC links in your region, enter the following command:

```
SHOW LINK TYPE=APPC
```

The following information about APPC links is displayed:

- Link Name
- Remote LU Name
- Link status
- Link type (parallel or single session)
- LU6.2 options supported (for example, mapping)
- Link and conversation level security supported
- The session limit for the link

Stop APPC Links

An APPC link can be stopped using the LINK STOP command.

Example

To stop the APPC link between SOLVE01 and NMA, enter the following command:

```
LINK TYPE=APPC STOP=NMA
```

Chapter 18: Broadcasts

This section contains the following topics:

Broadcast Services (see page 233)
List Broadcasts (see page 234)
Broadcast to Generic Resources (see page 235)
Set General Broadcasts (see page 238)
Set Primary Menu Broadcasts (see page 239)
Review Active Broadcasts (see page 239)
Send Dynamic Broadcasts (see page 240)
Receive Broadcasts (see page 241)
Access System Group Definition (see page 241)

Broadcast Services

Broadcast Services let you utilize the various broadcasting capabilities of your system services. Broadcast messages can be sent to terminals, users, and applications, and can be stored on a file.

Broadcast Services let you send the following types of broadcast:

- A general broadcast of one to four lines
- A primary menu broadcast of one line
- A broadcast of up to four lines to all, or specific, EASINET terminals
- A broadcast of up to four lines to all, or specific, system services attached terminals (including EASINET terminals)
- A broadcast of up to four hundred lines to all, or specific, users
- A broadcast of up to four hundred lines to a selected list of users.
- A broadcast of up to four lines to MAI users of an application (for example CICS or IMS).
- A broadcast of up to four lines to users of an NCL application.
- A broadcast to a specific user ID according to their preferred method of notification (as indicated in their UAMS security profile).

Note: Broadcast capabilities are also provided by the $BSCALL NCL interface. For more information, see the *Network Control Language Reference Guide*.

Types of Broadcasts

The broadcasts listed above can be grouped into the following types:

- General
- Primary Menu
- User

These broadcasts are either static or dynamic.

List Broadcasts

To display a list of all active broadcasts in the system

1. Enter **/BCAST** at the prompt.

   The Broadcast Services : Primary Menu appears.

2. Enter **L** at the prompt.

   The List Broadcasts panel appears.

   Note: The broadcasts in the selection list are either permanent or still to be viewed. A permanent broadcast is displayed until deleted by a user. Other broadcasts are displayed until they have been viewed by all target recipients.

View Active Broadcasts

To view the contents of an outstanding broadcast

1. Enter **S** (Browse) next to the required broadcast in the selection list.

   The details appear.

Delete Active Broadcasts

To delete an active broadcast

1. Enter **D** (Delete) next to the required broadcast in the selection list.

   The broadcast is deleted.

   Note: If you want to delete a broadcast before all target recipients have received the broadcast, enter **FD** (Force Delete) next to the required broadcast in the selection list.

Broadcast to Generic Resources

This section describes how to use the Broadcast Services facilities to enable broadcasts to regions belonging to a VTAM generic resource.

These facilities are available from the Broadcast Services : Primary Menu (**/BCAST**).

Enable Broadcasts to Generic Resources

To enable broadcasts to regions belonging to a VTAM generic resource, there must be a broadcast system group corresponding to that generic resource.

To add a broadcast system group corresponding to a generic resource

1. Enter **/BCAST** at the prompt.

   The Broadcast Services : Primary Menu appears.

2. Enter **LS** at the prompt.

   The Broadcast Services : Group List panel appears.

3. Press F4 (Add).

   The Broadcast Services : Group Definition panel appears.

4. Complete the following fields:

   Group Name

   Specifies the name of the new broadcast group.

   Description

   Describes the new broadcast group.

   Include Local System?

   Specifies whether the local system is included in the broadcast group. YES indicates that broadcasts can be issued on this system. NO indicates that broadcasts can be issued on remote systems only.

   Press F4 (Save)

   The group definition is saved. A message is returned, confirming that the new broadcast group has been added.

Add Resources to a Broadcast System Group

To add resources to a Broadcast System Group

1. From the Broadcast Services : Group Definition panel, press F5 (Resources).

   The Broadcast Services : Resource List panel appears.

2. Press F4 (Add).

   The Broadcast Services : Resource Definition panel appears.

3. Complete the following fields:

   Resource Type

   Specifies the type of resource to add. The following types of resource are supported:

   APPCLINK

   A predefined name of an APPC link between two regions.

   Limits: 12 characters

   DOMAIN

   A region (with a name of up to four characters) attached by INMC.

   Limits: 4 characters

   LU

   A network resource name that identifies the required region.

   Limits: 8 characters

   Note: When adding DOMAIN or LU resources, ensure that the necessary DEFLINK commands have been issued to allow the region to connect by using APPC.

   Resource Name

   Specifies the name of the resource to add. This name is used on the Broadcast Services : Send Menu when sending a broadcast to the system group.

   Press F3 (File).

   The changes are saved. The Broadcast Services : Resource List panel appears, with the new resource added.

Send Broadcasts to Generic Resources

When you have defined a broadcast system group for your generic resource, you can use that system group to send broadcasts to regions belonging to the generic resource.

To send broadcasts to generic resources

1. Enter **/BCAST** at the prompt.

   The Broadcast Services : Primary Menu appears.

2. Enter **S** at the prompt.

   The Broadcast Services : Send Menu appears.

3. Complete the following field:

   System Group

   Specifies the system group name for your generic resource.

   Enter the mnemonic of the type of broadcast that you want to send at the prompt.

   The broadcast is sent.

Set General Broadcasts

A general broadcast allows you to notify potential users of the system about critical events. For example, the impending unavailability of a major application such as production CICS or IMS subsystems.

A general broadcast can be up to four lines, and can be sent to all EASINET terminals or all system services terminals, including EASINET terminals. The lines of broadcast appear at the bottom of the terminal display and are available across system restarts.

When you send a general broadcast, the NCL variables &BROLINE1 to 4 are updated, and any panel containing these variables displays the text you have entered the next time those panels are accessed by a user.

To set a general broadcast

1. Enter **/BCAST** at the prompt.

   The Broadcast Services : Primary Menu appears.

2. Enter **G** at the prompt on the Broadcast Services : Primary Menu.

   The Broadcast Services : Edit Broadcast Text panel appears.

   On entry to this panel, the lines in the editor display the text from the last general broadcast that was issued. Details of the user who issued the last broadcast are also displayed.

3. Enter the text that you want to send as a general broadcast in the text lines provided and press F3 (File).

   The broadcast is saved.

   Note: This sets the broadcast so that it can be displayed, and saves the broadcast in the system services VFS file so it is available across system restarts.

Set Primary Menu Broadcasts

The primary menu broadcast allows you to set one line of text to display on the primary menu. This type of broadcast is useful for reminding users of something about the system they have logged on to. For example, you may want to remind users that the system is a test system and that they should not change anything.

To set a primary menu broadcast

1. Enter **/BCAST** at the prompt.

   The Broadcast Services : Primary Menu appears.

2. Enter **P** at the prompt.

   The Broadcast Services : Edit Broadcast Text panel appears. This looks similar to the edit panel for a general broadcast except there is only one text line available for input.

   When you first access this panel, details are given of when the text was last updated, and by which user ID.

3. Enter the text you want displayed on the primary menu in the text line, and press F3 (File).

   The broadcast is saved.

   Note: When you enter text in this line, the NCL variable &ZPMTEXT1, which is contained on the primary menu, is updated. When you press F3, the primary menu is updated and the broadcast is displayed when a user next accesses that menu. F3 also saves the broadcast so that it is displayed across system restarts.

Review Active Broadcasts

To review active dynamic broadcasts that are applicable to your terminal and user ID

1. Enter **/BCAST** at the prompt.

   The Broadcast Services : Primary Menu appears.

2. Enter **R** at the prompt.

   The Broadcast Services : Review Broadcasts panel appears. This panel displays all the broadcasts for your user ID and terminal in the same way you receive a broadcast when you sign on. To view subsequent broadcasts, press F8 (Forward).

   The dashed line above each broadcast on the review panel, includes a message which indicates whether the broadcast is deleted or whether it has been retained by the review function. Retained broadcasts can be deleted from the List Broadcasts panel only.

Send Dynamic Broadcasts

A dynamic broadcast allows you to send a message to a specific user, or to a range of users. When a broadcast is sent, a panel is displayed containing the broadcast lines and details of the broadcast initiation.

To send a dynamic broadcast

1. Enter **/BCAST** at the prompt.

   The Broadcast Services : Primary Menu appears.

2. Enter **S** at the prompt.

   The Broadcast Services : Send Menu appears.

   For information about the fields and options available on the prompt on the Broadcast Services : Send Menu, press F1 (Help).

Enter Broadcast Text

When you have selected the type of broadcast you want to send, and you have set the appropriate fields on the Send Menu, press Enter to invoke the Broadcast Services : Edit Broadcast Text panel. If you select option U, you can specify a maximum of 400 lines of broadcast text. If you select any of the other options, you can specify a maximum of four lines of broadcast text.

Note: If you specified option U and prompting in the Destination Mask field, you are presented with a list of user IDs. You must select the users you want to send the broadcast to and then press Enter to invoke the Broadcast Services : Edit Broadcast Text panel.

When you have entered the broadcast text, press F3 (File) to initiate the broadcast. An acknowledgment message is displayed when processing is complete. This message shows the total number of terminals and users that have received the broadcast at that point in time.

Note: If you send a broadcast to a user who is logged on more than once, the acknowledgment message includes each of their logons in the total number of user IDs that have received the broadcast. However, when the user has viewed the broadcast, the broadcast is discarded for their duplicate logons.

After a broadcast is sent, it is retained until you exit broadcast services. If you want to send another broadcast, the previous broadcast is displayed on the editor if you have not left broadcast services. This allows you to send the same broadcast again, or edit it to send to another user, or group of users.

Receive Broadcasts

When you receive a broadcast that has more lines than the screen has available, you can use the F8 (Forward) and F7 (Backward) keys to scroll through the broadcast.

To acknowledge receipt of the broadcast, you must press F3 (Exit). However, when you press F3 (Exit), you no longer have access to the broadcast. If the broadcast is more than one screen, you can press F3 (Exit) before you have read the whole broadcast.

Access System Group Definition

Broadcast services can be customized to send messages to one or more remote regions. This customization is performed by setting up system group definitions.

Each system group definition contains one or more remote regions to which a broadcast is sent. By specifying a system group definition when sending a broadcast, that broadcast is sent to the remote regions defined in that group definition. The local region can also be included in the system group definition.

The List System Groups option of the Broadcast Services primary menu is used to access the list of defined system groups on your system. From this list you can add, update, and copy system group definitions.

To access a list of the system groups defined

1. Enter **/BCAST.LS** at the prompt.

   The Broadcast Services : Group List panel appears.

   If no system groups have been defined, no entries appear in this list.

   For information about the information displayed and the actions available on the Broadcast Services : Group List panel, press F1 (Help).

Appendix A: Connecting Terminals

This section contains the following topics:

Supported Terminals (see page 243)
How Your Product Accepts Terminal Connections (see page 244)
Extended Attributes Support (see page 244)
Specify Screen Sizes (see page 245)
Specify Logmode Table Definitions (see page 246)

Supported Terminals

The IBM devices listed below are supported by your product region. Any device compatible with those listed is also supported.

IBM Device or Equivalent	Supported Functions
3270 Family LU0, LU2 (up to 255 x 255)	Full function
5550 (24 x 80)	Full function, including DBCS
3767 (or other LU-1 device)	OCS, printing

Information that allows your product to connect to these terminals is contained in their VTAM mode table definition.

Note: Your product region supports full operator functions through the operating system console by use of the MODIFY command. Privilege and authority available through the console can also be controlled. For more information about defining a system console, see the *Security Guide*.

How Your Product Accepts Terminal Connections

When a request for a terminal connection is made to your product, the following process occurs:

1. The logon exit is driven.
2. The logon exit examines the session parameters to determine the terminal type.
3. If the connection is accepted, the session parameters are modified, if necessary, and an OPNDST macro is issued. A BIND request then flows to the terminal. A CLSDST macro is issued if the request is rejected and information about the reason for the failure is sent to the terminal.
4. The terminal receives the BIND and if the parameters contained therein are suitable, logon proceeds and the session is established.

Extended Attributes Support

If you want to specify additional attributes for your terminal connections, you can specify a parameter in the logmode definition that allows your region to send a query to a requesting terminal.

The query obtains the following information from the requesting terminal:

- Extended (seven) color
- Extended highlighting
- Field outlining
- DBCS (double-byte data streams for special language support)
- Screen size

A query is sent when X‘80’ is specified in the second position of the PSERVIC field of the logmode definition, for example, PSERVIC=X‘0280...’.

Specify Screen Sizes

Some terminals and controllers support a special screen size definition in the logmode table definition called *unspecified viewport size*. When unspecified viewport size is specified, the screen size is determined from the parameters received from the terminal in a query.

If unspecified viewport size is not specified, the screen size for the terminal connection is determined by the default or alternate screen size for the terminal type. The default for all terminal types is 24 rows by 80 columns, and the alternate screen size is the largest screen size of which it is capable.

To enable unspecified viewport size, specify X‘03’ in the eleventh position of the PSERVIC field of the logmode table definition.

Notes:

- The session parameters are not altered to specify unspecified viewport size unless the definition also supports a query of extended attributes.
- The screen size for non-SNA terminals is always set using the information obtained by the query, regardless of the setting of the screen size information in the bind.
- An unspecified viewport size BIND fails if sent to a terminal that does not support it.
- The IBM 3174 controller supports an unspecified viewport size bind, as does a 3274 controller at Configuration Support D, Release level 65 and most distributed function devices.

Specify Logmode Table Definitions

The following table specifies the logmode table parameter values that are expected for different terminal types:

Terminal Type	Expected Parameter Values
LU01	FMPROF=X‘02’
LU02	FMPROF=X‘03’ PSERVIC=X‘01...’
LU03	FMPROF=X‘03’ PSERVIC=X‘02...’

Examples of suggested mode table entries for most 3270 panels can be found in the MODETABS member in the base installation library CC2DSAMP. In most cases, the default VTAM-supplied tables are adequate.

A sample mode table definition for use with LU-1 terminals that are to be supported by MAI-OC is also supplied in the member.

Note: For more information about coding mode table entries, see IBM's *Communications Server SNA Resource Definition Reference*.

Appendix B: Product Region JCL Parameters

This section contains the following topics:

Product Region JCL Parameter Descriptions (see page 247)
Product Name Keys (see page 254)

Product Region JCL Parameter Descriptions

The following list provides detailed descriptions of the JCL parameters that the started task member of your product region can specify. The default value is underlined.

ARMNAME=*name*

Specifies the name used to register a region with the Automatic Restart Manager (ARM). If not specified, the default is SVM_*acbname* where *acbname* is the ACB name set by the PRI= JCL parameter. Use the XOPT= JCL parameter to set ARM options.

DBCS={ NO | YES | IBM | FUJITSU }

Specifies the DBCS option.

DSNQLCL

Specifies the name of the local VSAM data set qualifier. This value is used when allocating data sets.

DSNQLNV

Specifies the name of the local non-VSAM data set qualifier. This value is used when allocating data sets.

DSNQSHR

Specifies the name of the shared VSAM data set qualifier. This value is used when allocating data sets.

DSNQSNV

Specifies the name of the shared non-VSAM data set qualifier. This value is used when allocating data sets.

DYNVVOL=*value*

Specifies a default volume for dynamically allocating files. If this parameter is not specified, no volume is specified in the dynamic allocations and SMS allocates the files according to its rules.

INIFILE=*filename*

Specifies the INI file to use for customization of parameters.

INIRESET

Specifies that all current parameters are to be ignored and the defaults used. You must not use this if an INI file is specified.

INIT={ NMINIT | *name* }

Specifies the name of the NCL procedure, in the COMMANDS procedure library, which is executed as part of system initialization before VTAM ACBs are opened.

Limits: *name* must be 1 to 8 characters.

INIWTO

Specifies that initialization messages are echoed to the system console.

INT=00/*nn*

Specifies which internal commands can be used. The parameter value comprises one hexadecimal byte, where each bit indicates acceptance of a particular command, as shown in the following table. The recommended value is INT=E4, that is, all commands shown in the table.

Bit	Command	Function
X'80'	SHOW STOR	Storage display
X'40	##AT, SH ##AT	AT-trap facility
X'20	##DP	Storage alter
X'10	(reserved)	
X'08	(reserved)	
X'04	##PMON	Performance monitor
X'02	(reserved)	
X'01	(reserved)	

NCLFM={ * | a }

(z/VM) Specifies the CMS filemode that applies to loading NCL procedures. This must be a valid CMS filemode.

NMDID=*domainid*

Defines the domain ID for this system. If the domain ID is not specified, the first characters (up to a maximum of 4) of the system primary ACB name, specified in the PRI parameter (or the default if not specified), are used as the default value.

The domain ID should be unique across all connected systems. If the value is not unique, it can produce naming conflicts, which restrict inter-system functionality, such as ROF sessions from background environments. Do not confuse this parameter with the system ID in the SYSTEMID parameter group.

Limits: One to four characters

NMSUP=*userprefix*

Specifies the System User Prefix that is used to prefix the user IDs for background system environments. If not specified, the domain ID is used. If the domain ID has not been specified, the first characters (up to a maximum of 4) of the system primary ACB name specified in the PRI parameter are used as the default value.

Limits: One to four characters

NOMODIFY

Specifies that communication between the operating system MODIFY facility and the product, using the system console, is not supported. If this operand is omitted, the use of MODIFY is supported.

NPF={ NO | YES }

Controls the use of NPF. Specify NO to inhibit NPF in the region. This is useful on test regions where security checking is not required but where a production UAMS is used.

NPFFM={ * | a }

(z/VM) Specifies the CMS filemode that applies when loading NPF tables. This must be a valid CMS filemode.

OPT=00/*nn*

Activates CPU accounting support. The parameter value comprises one hexadecimal byte, where each bit indicates the level of accounting, as shown in the following table, which lists the values of the OPT parameter.

Bit	Accounting
X'02'	CPU accounting for all threads
X'01	CPU accounting for NCL threads only

OSINP={ YES | NO }

Controls access to the OSCNTL data set. Access can be either read-only or update. Specifying YES (the default) indicates that the data set is opened for read only access. Specifying NO indicates that it is opened for update. When NO is specified, and an attempt to open the file for update fails, the region retries the open for read only access.

PRI={ NM | *acbname* }

Specifies the name of the primary VTAM ACB for the region to be used for terminal communication.

Limits: One to eight characters

Note: If this parameter is omitted, the default (NM) is also the default for the NMDID and NMSUP parameters, and the system ID in the SYSTEMID parameter group.

PROD={ *product* | (*product1,product2,...productn*) }

Specifies the list of product name key values for features to be included in the region. The PROD list implies that only the specified products are to be included in the region during initialization and that all other products are to be excluded.

Note: The list of included products cannot be modified without a restart of the region. Products not nominated in the PROD list are not resident in the region while it is active.

READY={ NMREADY | *membername* }

Specifies the name of the NCL procedure, in the COMMANDS procedure library, which executes as part of system initialization after VTAM ACBs have been opened successfully.

Limits: One to eight characters

SEC={ * | NO |PARTSAF |NMSAF |*name* }

Specifies whether the region uses a security exit.

If an asterisk (*) is specified, the region uses a security exit if one has been link edited into the NM001 load module. If no security exit has been link edited, then the region uses the NMUEX01 load module if it is in an accessible load library.

If NO is specified, no security exit is used. This overrides any link edited exit or the NMUEX01 load module.

If PARTSAF is specified, a vendor-supplied partial security exit that uses SAF is used.

If NMSAF is specified, the vendor-supplied security solution is used.

If name is specified, the named load module is loaded and used as the security exit. If this load module cannot be found, then the region terminates.

Important! If an abend occurs in the exit and the requested function cannot be performed, it is regarded as a security exposure and the region terminates. Message N00303 is sent to the console as a WTO, with RC=8.

Note: For more information about security exits, see the *Security Guide*.

SSID={ NO | * | *name* }

Specifies whether to establish a connection to a SOLVE Subsystem Interface (SSI) during initialization.

If NO is specified, then no connection to a SOLVE SSI is attempted. The connection is started (or attempted) only if the SSI parameter group requests a connection.

If an asterisk (*) is specified, then an SSID of the first four characters of your product region's job name is used.

If *name* is specified, then a specific SSID must be entered.

If * or *name* is specified, then an attempt to connect to the SSI is made immediately. If it fails, it retries every *n* seconds, depending on the value in the SSI parameter group.

TZ={ *shhmmm* | GMT }

Sets the internal time zone for your product region to a specific offset or Greenwich Mean Time (GMT). *The operating system offset is ignored, and no changes will be detected or processed.* Generally, you should not specify this parameter and your region runs on system local time.

The system hardware clock is usually set to GMT. The system local time is set by the operator when the operating system is initialized. The difference between the hardware time and the local time is the time zone offset from GMT.

When the hardware time is not set to GMT, you can use the TZ parameter to set this time. The region uses the system local time as the current time but uses the specified offset when calculating GMT.

s

Is plus (+) or minus (-), depending on whether the site is ahead (+) or behind (-) GMT.

hhmm

Specifies the four-digit hours and minutes value of the time zone offset.

Limits: 15 hours

If GMT is specified, then, assuming that the hardware clock is set to GMT, the internal time is set to GMT, rather than system local time. For example, if the system is four hours ahead of GMT and the local time is 5 pm, GMT time is 1 pm. Your region will use 1 pm as the displayed 'local' time.

UDBDEFER

Specifies that all UDB open processing is to be deferred and no attempts to open UDBs are to be made before the UDBCTL command is used.

NCL automatically attempts to open DD names that start with UDB on the assumption that they are utilized as UDBs. If specific processing options are required for UDBs, such as use of LSR pools, use the UDBCTL command to open the UDB with the required options.

Selecting DD names that do not start with UDB bypasses automatic open processing and requires opening by the UDBCTL command. Typically, UDBCTL commands to open UDBs are placed in the NMINIT or NMREADY procedures.

VFSENQ={ YES | NO }

Specifies whether the VFS data set is enqueued exclusively on initialization. The default is YES. NO blocks the ENQ and allows immediate restart.

When using the ABENDCMD command to restart a started task, z/OS do not hold the new task until the old one finishes dumping. This can cause errors during startup. For example, NCL dynamic allocation requests might fail if DISP=OLD is specified. Also, VSAM data sets might fail to open if SHROPTIONS 1 or 2 are in effect.

VSAMIO={ M | S | D }

Specifies how VSAM I/O is performed in your product region:

- M specifies that all VSAM requests are performed in the main task.
- S specifies that all VSAM requests are performed in the subtask.
- D specifies that your product dynamically switches between main task and subtask, based on load.

If significant VSAM activity is anticipated, specifying S improves processing overlap on multi-CPU machines.

WTO={ YES | NO }

Specifies whether the system console is to be sent monitor class messages if no definition exists for the system console user ID (usually *pppp*OPER).

XM={ TASK | ZIIP | BEST }

Specifies whether to move some processing performed by the main task of the region from the central processor (CP) to a zIIP.

- TASK or T specifies that main task processing occurs on the CP.
- ZIIP or Z specifies that main task processing occurs on a zIIP. If a zIIP is not available, an error message is generated and processing continues on the CP.
- BEST or B specifies that main task processing occurs on a zIIP if it is available; otherwise, processing occurs on the CP.

Default: TASK or T

XOPT={ *option* | *option, option, ...* }

Specifies the following options (for example, for dump processing and ARM registration). This operand can be specified more than once.

NOSXWEBU

(Default) Specifies that user security exit cannot handle web users.

SXWEBU

Specifies that user security exit has been modified to handle web users.

SDUMP

Specifies that write ABEND dumps to the SYS1.DUMP data set.

NOSDUMP

(Default) Specifies that send ABEND dumps to the normal dump data sets.

DAE

Specifies that DAE symptom information be provided when writing an ABEND dump.

NODAE

(Default) Specifies that DAE symptoms information is not provided.

ARM

Specifies that the region will try to register with the Automatic Restart Manager (ARM), using an ARM element name specified through the ARMNAME= JCL parameter. If a registered region fails, the sysplex ARM restarts that region automatically.

NOARM

(Default) Specifies that the region does not register with ARM.

RLSU

Requests VSAM Record Level Sharing (RLS) for the UAMS file.

NORLSU

(Default) Specifies that VSAM RLS is not used for the UAMS file.

PVLOAD

(Default) Loads persistent variables when starting a region.

NOPVLOAD

Prevents the loading of persistent variables when starting a region.

PWMIX

Specifies that mixed case passwords are supported.

If you enable this support, consider the following important points:

- Do not share a UAMS database with a region that does not support mixed case passwords.
- Ensure that all regions in a multisystem environment have this support enabled.

Product Name Keys

The following table shows product names with their associated product name keys and LMP codes for z/OS. It contains the following information:

Product Names

These reflect products that can be sold. Each product name has a product name key and one or more associated LMP codes (one for each operating system where the product is available).

Sub-functions

Many products include sub-functions, shown indented and bulleted. These sub-functions are not separately licensed, but the functionality of each required sub-function must be specifically included in the PROD= parameter.

This allows you to select the specific functionality that you require in a given region, and to exclude parts that you do not require.

Including Sub-functions

To include a sub-function, you must specify in the PROD= parameter the product name keys of both the product name and the sub-function.

Product Name	Product Name Key	LMP Code
CA NetMaster NM for SNA	SNA	YX
CA NetMaster NA	SNAAUTO	XY
NetView Connect	NVC	-
CA NetMaster NM for TCP/IP	TCPIP	Y7
ReportCenter	REPORTER	-
CA NetMaster SM for CICS	SOCKETMGMT	2D
WebCenter SDK	WEBCENTERSDK	-
CA NetMaster FTM	FT	X3
CONNECT:Direct Support	FTCD	-
CONNECT:Mailbox Support	FTMBX	-
FTP Support	FTFTP	-
CA XCOM Support	FTXCOM	-
ReportCenter	REPORTER	-
CA SOLVE:FTS Support	FTFTS	-
WebCenter SDK	WEBCENTERSDK	-
CA SOLVE:FTS	FTS	ZI
CA SOLVE:Operations Automation	OPSOS	ZX
WebCenter SDK	WEBCENTERSDK	-

Product Name	Product Name Key	LMP Code
CA SOLVE:Operations Automation for CICS	OPSCICS	Z5
WebCenter SDK	WEBCENTERSDK	-

Appendix C: SYSPARMS Operands

This section contains the following topics:

Generic SYSPARMS Summary Table (see page 257)
SYSPARMS Operand Descriptions (see page 264)

Generic SYSPARMS Summary Table

This following table provides a summary of SYSPARMS operands.

Note: After region initialization, you cannot use the SYSPARM command to change those operands that are set by parameter groups.

Some SYSPARMS operands apply to specific products only. These are indicated in the table by the following codes:

AS

Any product that uses Automation Services

SN

CA NetMaster NM for SNA

DS

CA SOLVE:FTS

SYSPARM	Description	Product Code
ACBRETRY	Specifies whether your product region attempts to re-open the VTAM ACB.	
AOMCUTOK	Sets the option for consoles to use SAF UTOKENS.	AS
AOMMLTO	Specifies the maximum time for a multiline WTO message to be completely collected.	AS
AOMPRFJI	Specifies whether the JES job ID prefixes the message text on an OCS panel.	AS
AOMPRFJN	Specifies whether the z/OS job name is to prefix the message text on an OCS panel.	AS
AOMPRFMN	Specifies whether the minor lines of multiline WTO messages have SYSCMD prefixes inserted.	AS

SYSPARM	Description	Product Code
AOMPRFSN	Specifies whether the originating system name prefixes the message text on an OCS panel.	AS
AOMPRFTM	Specifies whether the message text on an OCS panel will be prefixed with the time the message was issued.	AS
AOMSSID	Specifies the subsystem ID to be used by the SYSCMD facility.	AS
AOMTRACE	Specifies whether the message tracing is active.	AS
AOMTRCRC	Specifies how message routing codes are formatted.	AS
AOMTRLIM	Specifies the tracing limit for AOMTRACE.	AS
AUTOEXEC	Specifies whether your product region attempts to execute an unrecognized command string as an NCL procedure.	
AUTOREXX	Specifies whether a START command recognizes and executes a data set member as REXX procedure.	
CALLSHR0	Specifies whether a subtask shares subpool zero with the main task when an &CALL statement is executed.	
CDELAY	Specifies the time your product region waits before sending output to an OCS terminal when a user is entering input from the keyboard.	
CNMACBNM	Specifies the name of the CNM ACB.	SN
CONMSG	Specifies whether the message N07002 is written to the activity log each time a terminal connects to the system.	
DALDEFER	Specifies whether deferred mounting is requested when allocating data sets.	
DALRACF	Specifies whether automatic RACF protection is requested when dynamically allocating new data sets.	
DALRLSE	Specifies whether data sets created by dynamic allocation are defined with the RLSE option.	
DESC	Specifies the operating system description codes used for messages sent to the system console.	
DYNLMAX	Specifies the maximum number of dynamic INMC links that can be concurrently active.	
EDITCAPS	Specifies the default setting for the CAPS command.	
EDITNULL	Specifies the default setting for the NULLS command.	
EVCMDMIN	Specifies the minimum repeat frequency for the EVERY command.	

SYSPARM	Description	Product Code
FTSCPROC	Specifies the name of an NCL procedure that intercepts commands issued on completion of system transmissions.	DS
FTSFTM	Provides optional generation of additional data for some $$FTS events for use by CA NetMaster FTM.	
FTSMAXBK	Determines the maximum file block size that FTS processes.	DS
FTSRCDSN	Provides optional generation of an additional message (number N44807) at the end of a transmission, at the receiving system, that identifies the data set name into which the file has been received.	DS
FTSSMF	Specifies whether SMF records are generated by FTS on successful completion of transmission or receipt of a file.	DS
FTSTRDSN	Provides optional generation of an additional message (number N44307) at the end of a transmission, at the transmitting system that identifies the data set name just transmitted.	DS
HELDMSG	Specifies the default number of messages that are queued for an OCS window in HOLDING or AUTOHOLD mode, or if the window is closed.	
INMCBFSZ	Specifies the INMC buffer size for INMC traffic.	
INMCEX01	Defines the load module or phase name for the INMC primary security exit.	
INMCEX02	Defines the load module or phase name for the INMC secondary security exit.	
IPAMHB	Controls the use of heartbeats for TCP/IP INMC and APPC links.	
JRNLPROC	Specifies the NCL procedure to start when a journal swap occurs.	
JRNLSWAP	Indicates whether the NDB journal data set is automatically swapped if a space error occurs on the active journal.	
LANG	Specifies the language code for the system.	
LMSGWARN	Specifies the repeat frequency at which OCS operators are warned of lost messages.	
LNKTRACE	Specifies whether a trace message is issued each time an attempt to open a session to a remote region fails.	
LOCKPROC	Specifies the procedure that is invoked when a LOCK command is issued.	
LOGPAGE	Specifies the number of lines the activity log has on each page.	
MAIACBOR	Specifies the maximum number of retries permitted when opening an MAI ACB.	
MAIEX02	Specifies the name of an exit routine to take control whenever an MAI-OC session is started or ended.	

SYSPARM	Description	Product Code
MAIEX02S	Specifies the way MAI-OC serializes calls to the MAIEX02 exit routine.	
MAIONL	Specifies whether MAI appends a new line character (X'15') to data sent to the target application (that is, inbound from the terminal).	
MAIOPREF	Specifies a 1- to 5-character string, which is used as the prefix to an LU name generated by MAI-OC.	
MAIOTRNS	Specifies the translate table used by MAI-OC.	
MAXRUSZ	Specifies the maximum request unit size for APPC sessions.	
MENULU1	Specifies an alternate soft menu for LU1 logons.	
MENUPROC	Specifies an alternate primary menu procedure name.	
MODLUSER	Specifies the name of a UAMS user ID definition, present on the UAMS data set, to use as a model for dynamic user generation.	
NCLEX01	Specifies the NCL general authorization exit.	
NCLGBTRC	Specifies a single global variable name, or a generic global variable prefix, that is to be traced as changes occur.	
NCLOGTRM	Specifies whether NCL writes log message N03906 on completion of each NCL procedure.	
NCLTRLFF	Specifies how many X'FF' field separators NCL places at the end of a record written to a UDB.	
NCLTRMAX	Specifies the number of NCL trace messages that are generated when an NCL procedure is invoked.	
NDBLOGSZ	Sets the number of VSAM logical records that are formatted as a journaling area when an NDB is created using the NDB CREATE command.	
NDBOPENX	Controls whether the nominated NCLEX01 is called for &NDBOPEN.	
NDBPHONX	Registers the name of the NCL phonetic exit program.	
NDBRUMIN	Sets the minimum adjacent record ID (RID) range to be re-used.	
NDBRUSCP	Sets the percentage of used RID space to be scanned for re-use.	
NDBSCANO	Enables or disables the scan optimizer.	
NDBSUBMN	Sets the minimum number of subthreads that stay active, for any NDB, awaiting database requests that can run asynchronously.	
NDBSUBMX	Sets the maximum number of subthreads permitted.	
NONSWAP	(z/OS) Specifies whether this system is to run non-swappable or swappable.	

SYSPARM	Description	Product Code
NRDLIM	Specifies the number of non-roll delete messages that the system queues before deleting the oldest messages.	
NSPRTINT	Sets the default timeout for solicited responses to commands sent to CA NetSpy.	
NTSACCT	Specifies whether NTS accounting data is collected for selected sessions only, for all sessions, or for no sessions.	SN
NTSCINTV	Specifies the NTS correlation interval value.	SN
NTSCLOSE	Specifies whether any sessions, which remain active when NTS is closed by system initialization, are to be considered ended for the purpose or output logging.	SN
NTSCNMQ	Specifies whether NTS queues NTS CNM requests.	SN
NTSEVENT	Controls NTS event generation.	SN
NTSINTSV	Specifies whether NTS intensive message logging is active.	SN
NTSMAIEX	Specifies whether MAI sessions are presented to the NTS user exit.	SN
NTSMAISV	Requests that MAI inform NTS of any current MAI session and subsequent MAI sessions.	SN
NTSMAXTP	Identifies the maximum number of trace PIUs that can be specified by a class definition or operator command.	SN
NTSMAXTR	Identifies the maximum number of resources that can have a specific STRACE request outstanding.	SN
NTSRSINT	Specifies the interval length for collecting resource statistics.	SN
NTSRSLIM	Specifies how many NTSRSINT intervals can occur before the statistics collected for the oldest interval are overwritten.	SN
NTSRSTAT	Specifies whether resource statistics are collected.	SN
NTSSAWBF	Specifies the number range and size of the buffers allocated by VTAM for collecting SAW data.	SN
NTSSKEEP	Specifies the default session keep count for sessions written to the NTS database.	SN
NTSSMFTM	Specifies whether to cause CA NetMaster products to write timestamps in type 39 SMF records in local time or GMT.	SN
NTSTRBFX	Specifies whether the trace final queue buffers are consolidated when the first wrap occurs.	SN
NTSTRCBF	Specifies the number range and size of the buffers to be allocated by VTAM for collecting trace data.	SN

SYSPARM	Description	Product Code
OCSHLITE	Specifies the type of highlighting to use for messages appearing in OCS windows.	
OCSTIME	Specifies whether the time appears at the end of the title line of an OCS window.	
PANLBFSZ	Specifies the maximum outbound data stream size that can be generated for any terminal attached to your product region.	
PANLBUFF	Specifies the maximum number of pages of virtual storage that can be used for concurrent terminal output operations.	
PPOCOLOR	Specifies the color to use when displaying unsolicited VTAM (PPO) messages on OCS consoles.	
PPOHLITE	Specifies the type of highlighting used when displaying unsolicited VTAM messages on OCS consoles.	
PPOSOMSG	Specifies how PPO messages received from VTAM, as the result of VTAM commands entered from a system console or a local OCS window, are logged.	
PPOUSMSG	Specifies how unsolicited PPO messages are written to the activity log, regardless of DEFMSG options.	
PWEXPIRE	Specifies the number of days after which users are forced to change their password.	
PWMAX	Specifies the maximum acceptable length for passwords.	
PWMIN	Specifies the minimum acceptable length for passwords.	
PWRETRY	Specifies the number of times an incorrect password is accepted before a logon attempt is denied.	
ROUTCDE	Specifies the operating system routing codes to use for unsolicited messages sent to the system console.	
RXCSTACK	Specifies the working stack size (in KB) of the REXX compiler.	
RXISTG	Specifies the size (in KB) of the initial storage to get for a REXX process.	
RXMCSZ	Limits the maximum size (in KB) of a compiled REXX object.	
RXMSTG	Specifies the maximum amount of storage (in KB) a REXX process can use.	
RXQSFIX	Specifies whether to handle quoted string in REXX that has crossed multiple lines.	
RXRETAIN	Specifies the size (in KB) of the REXX retained procedure pool.	
RXXSTACK	Specifies the working stack size (in KB) of a REXX process.	

SYSPARM	Description	Product Code
SESSMSG	Specifies whether trace message N35007 is issued each time a session to a remote system opens or fails.	
SMFID	Specifies the SMF record identifier to use in the generation of SMF records.	
STGWRN	Specifies the number of kilobytes below the 16 megabyte line at which a N01801 message is issued as a WTO indicating that the storage thresholds have been exceeded.	
STGWRNXA	Specifies the number of kilobytes above the 16 megabyte line at which a N01801 message is issued as a WTO indicating that the storage thresholds have been exceeded.	
SYSCONMU	Specifies the default user ID for a master console user when it is not signed on.	
SYSCONNM	(z/OS) Specifies the LU name that is assigned to system console environments.	
SYSCONSO	(z/OS) Specifies whether the console user ID can default and whether signon is required.	
SYSCONUI	Specifies the default system console signon name.	
SYSCONXU	(z/OS) Specifies whether external console user IDs are used when signing on consoles.	
SYSLOG	(z/OS)Specifies whether none, all, or unsolicited VTAM messages written to the activity log, are to be written to the system log.	
SYSLOGFM	(z/OS) Specifies the format of the SYSLOG lines.	
TNDSREG	Specifies whether the Telnet server registers new connections with the Data Space Manager.	
TRACEOPT	Specifies the trace options to apply when tracing data streams sent to or from a terminal.	
USERPW	Specifies whether the NCL system variable &USERPW is available for use in MAI logon data.	
VDISPLAY	Specifies how the VTAM display command (D) is processed for users with command network partitioning.	
VTAMID	Specifies the system procedure name used for starting VTAM.	
XABELOW	Specifies whether your product region allocates buffer storage below the 16-MB line in XA systems if all storage in the extended private area has been used.	

SYSPARMS Operand Descriptions

ACBRETRY={ NO | YES }

Indicates whether your product region attempts to reopen the VTAM ACB. If YES, the system attempts to open the VTAM ACB during initialization. The system also attempts to reopen the ACB if it is closed at some time during normal processing (for example, if VTAM is shut down), at 30 second intervals.

Default: YES

AOMCUTOK={ NO | YES }

Sets the option for consoles to use SAF UTOKENs. This requires use of a user security exit that returns UTOKENs.

Note: UTOKEN usage is supported for both JES and EXTMCS consoles.

This operand cannot be changed when AOM is active or when consoles are acquired.

Default: YES

AOMMLTO=*number*

Specifies the maximum time (in seconds) that the SYSCMD facility will allow for any one multiline WTO to be completely collected. If this time-out value is exceeded, and the end line has not yet been seen, the multiline WTO is sent through as is for further processing by AOMPROC, and so on.

Default: 5

Limits: 2 to 60

AOMPRFJI={ NO | YES }

Specifies if the JES job ID is to prefix the message text when displayed at an OCS screen, for messages sourced by an z/OS system. The value of this parameter is used to set the message prefixing actions when a new OCS or NCL &INTCMD environment is created.

Default: YES

AOMPRFJN={ NO | YES }

Specifies if the z/OS job name is to prefix the message text when displayed at an OCS screen, for messages sourced by an z/OS system. The value of this parameter is used to set the message prefixing actions when a new OCS or NCL &INTCMD environment is created.

Default: NO

AOMPRFMN={ NO | YES }

Specifies if the minor lines of z/OS-sourced multiline WTO messages are to have SYSCMD prefixes inserted when displayed. The value of this parameter is used to set the message prefixing actions when a new OCS or NCL &INTCMD environment is created.

Default: NO

AOMPRFSN={ NO | YES }

Specifies whether the originating system name is to be prefixed to the message when it is delivered to an OCS window. The value of this parameter is used to set the message prefixing actions when a new OCS or NCL &INTCMD environment is created.

Default: NO

AOMPRFTM={ NO | YES | HMS | HM }

Specifies if the time the message was issued is to prefix the message text when displayed at an OCS screen. The value of this parameter is used to set the message prefixing actions when a new OCS or NCL &INTCMD environment is created.

AOMPRFTM=YES or AOMPRFTM=HMS causes the time to be displayed as *hh:mm:ss*.

AOMPRFTM=HM causes the time to be displayed as *hh:mm*.

Default: YES

AOMSSID={ * | *sub-system-id* }

Specifies the subsystem ID to be used by SYSCMD facility. The asterisk (*) uses the first four characters of the job name or started task (STC) name.

Default: NETM

AOMTRACE={ NO | YES }

Specifies whether the message tracing is to be active (for the number of messages defined in AOMTRLIM).

If YES, all messages from the local operating system are logged with trace information.

Default: NO

AOMTRCRC={ HEX | LIST}

Specifies, for messages traced by AOMTRACE, how the message routing codes are to be formatted.

AOMTRCRC=HEX (the default) displays the routing codes in HEX format, showing 32 hex digits (corresponding to 16 bytes, or 128 bits, of routing codes).

AOMTRCRC=LIST displays the routing codes in list format, with numbers or ranges of numbers indicating the routing codes.

Default: HEX

AOMTRLIM=*number*

Specifies the tracing limit. When reached, AOMTRACE is set to NO.

Default: 100

Limits: 1 to 100

AUTOEXEC={ NO | YES }

If an unrecognized command string is entered from an OCS terminal, this operand specifies whether your product region assumes it is the name of an NCL procedure which it then attempts to execute.

If AUTOEXEC=NO is specified, unrecognized strings are rejected with an error message.

If AUTOEXEC=YES is specified and the string is a valid potential NCL procedure name (that is, a valid member name), your product region automatically generates a START command for commands entered from OCS, or an EXEC command for commands issued by an NCL procedure. If the string is not the name of a defined NCL procedure the command is then rejected.

Default: YES

AUTOREXX={ NO | YES }

Specifies whether a START command recognizes and executes a data set member as REXX procedure:

- YES executes members as REXX procedures.
- NO executes members as NCL procedures.

Default: NO

CALLSHR0={ NO | YES }

Specifies whether subpool zero is to be shared between &CALL subtasks and your product region's main task.

When &CALL statements are executed, a subtask is attached to execute the target load module. By default, the subtask does not share subpool zero with the main task. However, if multiple called modules open and process the same VSAM data set, VSAM can suffer abends unless subpool zero is shared.

Default: NO

CDELAY=*number*

Specifies the time (in seconds) your product region waits before sending output to an OCS terminal when a user is entering input on the keyboard.

OCS terminals can receive unsolicited output at any time. If a user is entering input on the keyboard and output becomes available, a contention condition arises (on SNA terminals) which prevents a terminal from accepting output until input is complete.

This contention is broken if it continues for more than the CDELAY interval by interrupting the keyboard input and forcing the available output to the terminal. Therefore, the CDELAY parameter should be set to an interval that provides a reasonable period for uninterrupted entry of commands on the keyboard without causing excessive delay before output can be sent. The default value of 15 seconds is usually adequate. CDELAY also applies to NCL panel display.

Default: 15

Limits: 0 to 300

CNMACBNM=*cnm_acb_name*

Specifies the name of the CNM ACB.

Limits: One to eight characters

CONMSG={ NO | YES }

Specifies whether your product region is to write message N07002 to the activity log each time a terminal is connected to the system.

If using EASINET, this message occurs each time a terminal is returned to EASINET control after being logged on to another application. The LOGPROC NCL procedure can intercept these messages and retain statistics on terminal usage and the length of time terminals remain connected to other applications.

Default: NO

DALDEFER={ NO | YES }

Specifies whether deferred mounting is to be requested when allocating data sets.

Default: NO

DALRACF={ NO | YES }

Specifies whether automatic RACF resource protection is requested when dynamically allocating new data sets.

Default: NO

DALRLSE={ NO | YES }

Specifies whether data sets created by dynamic allocation are defined with the RLSE option to free unused secondary extents.

Default: NO

DESC={ (*n,n,n*) | NONE }

Defines the operating system descriptor codes used for messages sent to the system console, including messages associated with the *pppp*OPER user ID. Specify NONE to cancel any previously set description codes.

Note: For an explanation of the impact of specifying certain codes, see the appropriate operating system manual.

Default: NONE

Limits: 1 to 16 and NONE

DYNLMAX=*number*

Specifies the maximum number of dynamic INMC links that can be active at the same time. Enter 0 to prevent any dynamic links from being established and to disable the dynamic facility.

Default: 10

EDITCAPS={ OFF | ON }

Specifies the default setting for the CAPS command when using Panel Services. OFF specifies that entered data is to be retained in upper and lower case format. ON specifies that entered data is to be converted to upper case. The default attribute can be changed during editing by the CAPS command. When a member is saved, Edit Services retains the current CAPS setting and reinstated that value when the member is next edited regardless of the setting of the EDITCAPS operand.

Default: OFF

EDITNULL={ ON | OFF }

Specifies the default setting for the NULLS command when using Panel Services. If OFF is specified, trailing blanks are retained on each edit line. If ON is specified, trailing blanks on each edit line are converted to nulls (X'00') before being displayed for editing. Using nulls on an edit line lets you use the insert key to insert data amongst other text on the line. Use the NULLS command to change the default attribute while editing.

When a member is saved, Panel Services retains the current NULLS setting and reinstates that value when the member is next edited, regardless of the setting for the EDITNULL operand.

Default: ON

EVCMDMIN=*number*

Determines the minimum repeat period (in seconds) permitted for an EVERY command. This can be used to stop the system from being flooded with timer commands inadvertently. If a value of 0 is specified, an EVERY command is executed immediately.

Default: 10

Limits: 0 to 300

FTSCPROC={ *procname* | NONE }

Specifies the name of an NCL procedure that will intercept commands issued on completion of system transmissions.

These commands are executed under the control of the background system (BSYS) environment, subject to BSYS's authorization level. If a greater degree of control over these commands is required, you can write an NCL procedure (an FTSCPROC) that is called whenever a terminating transmission attempts to issue a command. The command, along with information about the terminating transmission, is passed to your FTSCPROC, which may choose to issue the command or suppress it. To activate your FTSCPROC, set its name in this operand. If you issue SYSPARMS FTSCPROC=NONE, no FTSCPROC will be used, and commands will be executed directly.

Note: For more information, see the *Administration Guide*.

Default: NONE

FTSFTM={ YES | NO }

Specifies whether extended event data is created for $$FTS events for use by CA NetMaster FTM.

Note: For more information, see the *Network Control Language Reference Guide*.

Default: NO

FTSMAXBK={ 32K / 64K }

Determines the maximum file block size that CA SOLVE:FTS will process.

In some circumstances (for example, tape data sets created with block sizes greater than 32 KB), it may be necessary to set this operand to 64K to process these files successfully.

Default: 32K

FTSRCDSN={ NO | SYS | ALL | LOG }

Provides optional generation of an additional message (number N44807) at the end of a transmission, at the receiving system, that identifies the data set name into which the file has been received.

- SYS specifies that the extra message is generated only for SYSTEM transmissions.
- ALL specifies that the message is generated for all transmissions.
- LOG specifies that the message is generated for all transmissions but is written to the activity log only.

For SYS or ALL, the message is written both to the activity log and to OCS operators profiled to receive FTS messages. The generation of this extra message enables the system's LOGPROC NCL procedure to monitor the successful completion of transmissions and to dynamically generate JCL job streams that can be submitted to process the data set just received.

Default: NO

FTSSMF={ NO | YES }

Specifies whether SMF records are to be generated by CA SOLVE:FTS on successful completion of transmission or receipt of a file.

Generation of SMF records is possible only if CA SOLVE:FTS has been assigned an SMF record identifier to be used on all SMF records generated by the system. This identifier is set using the SMF parameter group of the Customizer.

Default: NO

FTSTRDSN={ NO | SYS | ALL | LOG }

Provides optional generation of an additional message (number N44307) at the end of a transmission, at the transmitting system that identifies the data set name just transmitted.

- SYS specifies that the extra message is generated only for SYSTEM transmissions.
- ALL specifies that the message is generated for all transmissions.
- LOG specifies that the message is generated for all transmissions but is written to the activity log only.

For SYS or ALL the message is written both to the activity log and to OCS operators profiled to receive CA SOLVE:FTS messages.

Default: NO

HELDMSG=(*xxx,yyy*)

xxx defines the default number of messages that are queued for an OCS window in HOLDING or AUTOHOLD mode, or where the window is closed. When this limit is reached, the earliest messages are discarded to allow the latest messages to be queued and a warning that messages are lost is sent to the terminal. *yyy* defines the maximum depth of the message queue that any user can request when using the PROFILE HOLD command. This lets you limit the size of any individual operator's message queue by overriding the default range of the PROFILE HOLD command.

Default: (200,999)

Limits: 10 to 999, and *xxx* cannot exceed *yyy*.

INMCBFSZ=*number*

Specifies the INMC buffer size (in KB) for all INMC traffic. This buffer size is the default size used for all outbound messages on all INMC sessions except for those sessions whose bind specifies a maximum RU size smaller than INMBFSZ. For these sessions the message size is equal to or smaller than the bind RU size.

Increasing INMCBFSZ can improve performance on high speed links, such as channel-to-channel or microwave links. Slow links can benefit from a lower INMCBFSZ value.

This operand can be changed at any time. However, the changed value of INMCBFSZ is not reflected in active links until they are stopped, reset, and restarted.

Default: 4

INMCEX01={ *exitname* | NONE }

Defines the load module for the INMC Primary Security Exit to be invoked whenever any INMC link becomes active. *exitname* is any valid module name. If you specify NONE at a later stage, the definition of any existing primary exit is canceled. This operand is applicable only for systems configured with INMC.

Default: NONE

INMCEX02={ *exitname* | NONE }

Defines the load module for the INMC Secondary Security Exit to be invoked whenever any INMC link becomes active. *exitname* is any valid module name. If you specify NONE at a later stage, the definition of any existing secondary exit is canceled. This operand is applicable only for systems configured with INMC.

Default: NONE

IPAMHB={ NO | (a,b) }

Controls the use of heartbeats for TCP/IP INMC and APPC links.

IPAMHB=NO disables heartbeats for INMC/APPC links using TCP/IP (regardless of the setting on the other side).

IPAMHB=(*a*,*b*) sets up heartbeats as follows:

- *a* is the heartbeat send interval, in seconds. The valid range is 10 to 100.
- *b* is the heartbeat loss toleration count. The valid range is 2 to 5.

If no heartbeats are received in (*a***b*) seconds from the other side of the session, then the session is closed. For INMC, the link is retried later.

If both sides of a link want heartbeats, then the *minimum* of each value is used; that is, the minimum of the two send intervals and the minimum of the two loss toleration counts.

Default: NO

JRNLPROC=*procname*

Specifies the NCL procedure to be started when a journal swap occurs.

Default: $NDJPROC

JRNLSWAP={ NO | YES }

Specifies whether the NDB journal data set is to be swapped automatically if a space error occurs on the active journal.

When YES is specified, the journal is swapped to the alternate journal if the file full condition occurs.

When NO is specified, the journal error prevents further updates to journaled NDBs and any updates in progress are held over until the NDB is successfully restarted.

Default: YES

LANG={ US | UK | *cc* }

Defines the language code for the system and the default language code for users.

Note: For the system recognized values that can be used to replace the *cc* option on this operand, see the *Network Control Language Reference Guide*.

Default: US

LMSGWARN=*number*

Specifies the repeat frequency at which OCS operators are warned of lost messages when in HOLDING, AUTOHOLD, or FS-HOLD mode. The warning message appears every *number* messages lost.

Default: 10

Limits: 1 to 999

LNKTRACE={ NO | YES }

Specifies whether a trace message is to be issued each time an attempt to open a session to a remote region fails.

If YES is specified, the system issues an N35006 trace message to Monitor status users each time an attempt to open a session to a remote system fails for any reason. The system retries such failures indefinitely at 60 second intervals until the session is opened or the link is stopped. If a remote system cannot be contacted, turn on this trace and examine the content of the LNKTRACE message to help resolve the cause of failure.

Default: NO

LOCKPROC=*procname*

Specifies the LOCK command procedure. This procedure is invoked when a user enters the LOCK command to lock the terminal.

Default: $NMLOCK (supplied procedure)

LOGPAGE=*number*

Defines the number of lines per page of the activity log. This operand must be executed near the start of the NMINIT initialization procedure to become immediately effective. Values of 30 to 250 can be specified.

Default: 60

MAIACBOR=*nn*

An MAI ACB name is constructed using a prefix and a numeric suffix. When an ACB fails to open, the suffix is incremented and the ACB open is retried. There are two types of failure:

- A failure because the ACB is in use or inactive (non-error situation)
- A failure for any other reason (error situation)

There is no limit set on the number of retries allowed for non-error failures.

This operand sets the maximum number of consecutive retries allowed when an MAI ACB fails to open in an error situation.

When an ACB fails to open in a non-error situation, the retry limit is reset to the value of this operand.

Default: 5

Limits: 5 to 99

MAIEX02={ *exitname* | NO }

Specifies the name of an exit routine to take control whenever an MAI-OC session (both MAI-OC and MAI-FS) is started or ended and optionally when the VTAM ACB has been opened. The exit routine also receives notification of the LU name chosen by MAI. This routine is given information about the session to be created and can validate and alter those details, as well as having the capability of refusing the session. In addition, the routine has the capability of correlating information across session start and end calls. The MAIEX02 parameter supplies the name of the user exit routine to perform this function. MAIEX02=NO indicates that an existing exit is to be disabled or that no exit is required.

MAIEX02S={ SYSTEM | USER }

Specifies the way in which MAI serializes calls to the MAIEX02 exit routine. MAIEX02S=SYSTEM, the default, ensures that all calls to the routine are, in effect, queued within any one system, while MAIEX02S=USER only queues concurrent calls at an individual user level.

MAIONL={ NO | YES }

Specifies whether or not MAI will append a new line character (X'15') to data sent to the target application (that is, inbound from the terminal).

SYSPARMS MAIONL=YES, the default, appends a new line; SYSPARMS MAIONL=NO does not.

Default: YES

MAIOPREF=*prefix*

Specifies a character string which is used as the prefix to an LU name generated by MAI-OC, for example:

```
SYSPARMS MAIOPREF=MAI02
```

If the SYSPARMS MAIOPREF operand is not coded, MAI-OC assumes a default prefix of NMMAV.

Limits: One to five characters

MAIOTRNS=(*xx,yy*)

When data is received from an application across an MAI-OC session, MAI-OC translates, by default, any characters below X'40' to an underscore (_) before displaying it at an OCS window and leaves all other data intact. The MAIOTRNS operand can be used to alter the translate table used by MAI-OC in this process. The value *xx* is two hexadecimal digits representing a value to be translated and *yy* two hexadecimal digits representing a value into which *xx* is to be translated. The *yy* value must represent a *printable* character.

For example, to translate a hexadecimal zero (X'00') to a blank, specify:

```
MAIOTRNS=(00,40)
```

MAXRUSZ=*number*

Specifies the maximum request unit size for APPC sessions. If necessary this value overrides the LU6.2 session BIND parameters. The value specified on this command applies to new sessions established after the command is complete. The default value is 4096. The value specified is converted to a single byte RU size of the form X'*ab*' where number=*a**2***b*. Values must be in the range 128 to 32768 and the values which cannot be converted directly are rounded down.

MENULU1=*procname*

Specifies an alternate menu for LU1 logons.

Default: $NMPMLU1 (supplied procedure)

MENUPROC=*procname*

Specifies an alternate primary menu procedure name.

Default: $NMPMENU (supplied primary menu procedure)

MODLUSER={ *userid* | NONE }

Specifies the name of a UAMS user ID definition, present on the UAMS data set, to be used as the model for dynamic user ID generation. Use the following scenarios to determine the most appropriate action for your system:

- If you do not want to specify a model for dynamic user ID, specify NONE or let it default.
- If this operand is coded and a user attempts to log on to a product region with an unknown user ID, but their password is that associated with the model user ID, the system automatically creates a new user ID for the unknown user. This ID has exactly the same attributes and privileges as the model user ID.
- If a password security exit is implemented, the exit can alter the model name that is to be used.
- If a full security exit is implemented this operand is ignored.

Default: NONE

NCLEX01={ *exitname* | NO }

Specifies the name of the NCL authorization exit. This exit provides security for NCL procedures and &SMFWRITE verbs.

Note: For full information about NCLEX01, see the *Security Guide*.

If NCLEX01 is frequently invoked, use the LOAD MOD=*exit_name* command to load a re-entrant version of the exit into the region to eliminate the overhead of loading the exit for each authorization.

If NO is specified, then the current exit name is deleted and exit invocation is stopped.

Note: The NCLEX01 load module is executed under a subtask and can therefore issue I/O and WAIT operations without impacting the main system.

NCLGBTRC= { *name* | *prefix }**

Specifies a single global variable name, or a generic global variable prefix that is to be traced as changes occur to them. Each time an assignment occurs into a traced global variable, a log record is written that identifies the process performing the assignment. The first eight bytes of data are also traced.

If *name* is specified, only one global variable is traced. If *prefix* is specified, all global variables that start with the nominated prefix, excluding the standard global variable prefix, are traced as their values change.

The trace record is written as a single message, N23312, to the activity log. Tracing is turned off if no name or prefix is specified.

NCLOGTRM={ NO | YES }

Specifies whether NCL writes a log message on completion of each NCL procedure. If YES is specified, the log message provides statistics on NCL processing units used by the procedure.

Default: NO

NCLTRLFF={ ONE | MULT }

Specifies how many X'FF' field separators NCL places at the end of a record written to a UDB. If MULT is specified, and a record is written to a UDB where the record contains multiple null variables at the end, one X'FF' field separator is appended to the UDB record for each 'null' variable on the &FILEPUT or &FILEADD statement.

If NCL-format UDBs are created for processing by external systems, the use of this operand should be consistent.

Default: ONE

NCLTRMAX=*number*

Specifies the maximum number of NCL trace messages that are generated in any one invocation of an NCL procedure. If 0 is specified, all tracing is inhibited.

Default: 100

Limits: 0 to 9999

NDBLOGSZ=*n*

Sets the number of VSAM logical records that will be formatted as a journaling area when an NDB is created using the NDB CREATE command. This journal area provides transaction integrity across system failures. If the LOGSIZE parameter is specified on NDB CREATE, it overrides this value.

This value can be changed prior to issuing an NDB CREATE command, to change the journal size for that database.

Journal size is influenced by the possible complexity of an add, update, or delete operation on the database, which in turn depends on such things as the size of the data record, and the number of keys being added. The journal automatically extends if it is under-allocated.

Note: For more information, see the *Network Control Language Programmer Guide*.

Default: 40

Limits: 10 to 200

NDBOPENX={ NO| YES }

Controls whether the nominated NCLEX01 is called for &NDBOPEN.

NDBPHONX=*name*

Registers the name of the NCL phonetic exit program.

NDBRUMIN=*nnn*

Sets the minimum number of consecutive RID numbers that can be reused. You can change this operand at any time.

Default: 20

Limits: 10 to 100

NDBRUSCP=*nnn*

Sets the percentage of RID space to be scanned to collect RIDs for reuse. Values of 95 or greater cause the complete NDB to be scanned. For an NDB that is scanned daily, a default value of 15 means that the NDB is scanned completely every week. You can change this operand at any time.

Default: 15

Limits: 5 to 100

NDBSCANO= { NO | YES }

Enables (YES) or disables (NO) the scan optimizer.

Notes:

- The setting of the NDBSCANO value does not affect NDBs that are already started at the time the command was issued.
- An individual NDB can override the setting of the NDBSCANO command using the OPTIMIZE operand of the NDB START command.

Default: YES

NDBSUBMN=*n*

Sets the minimum number of subthreads that will stay active, for any NDB, awaiting database requests that can run asynchronously (&NDBSCAN and &NDBGET requests). When a database request arrives that can be run asynchronously, the database handler starts a separate copy of itself to run that request, unless there are already NDBSUBMX subthreads running. As the subthreads run out of work, they terminate unless the NDBSUBMN limit is reached.

Default: 3

Limits: 0 to 20

NDBSUBMX=*n*

Sets the maximum number of subthreads allowed. See NDBSUBMN.

Default: 5

Limits: 1 to 20

NONSWAP={ YES | NO }

Specifies whether this system is to run non-swappable (YES), or swappable (NO). This operand is valid only if your product region is running authorized.

For z/OS and MSP, your product region makes itself non-swappable automatically before system initialization. The NONSWAP operand can be utilized to change this status either during initialization or at any time after.

For VOS3, if your product region is authorized, it runs swappable, by default. It can be changed to run non-swappable by specifying YES. However, once running non-swappable, it is not possible to change the status back to swappable without stopping and restarting the system.

Note: For z/OS and MSP, optional features which require non-swappable operation blocks attempt to revert to swappable operation.

NRDLIM=*number*

Specifies the maximum number of NRD messages that the system queues at any time before discarding the oldest messages. This queue is used by the NRDRET command to refresh the OCS NRD message display.

Default: 200

Limits: 10 to 10000

NSPRTINT=*interval*

Sets the interval between retries when trying to connect to a CA NetSpy region. It is an integer number of seconds.

Default: 30

Limits: 10 to 600

NTSACCT={ SELECTIVE | ALL | NO }

Specifies whether NTS accounting data is to be collected for selected sessions only, for all sessions, or for no sessions. NTSACCT=SELECTIVE means that NTS attempts to collect accounting data for those sessions which have a *sawclass* specifying ACCT=YES (see the DEFCLASS command). NTS starts a specific trace for the primary resource if possible, otherwise the secondary resource if possible, to gather accounting information. No attempt is made to stop such selective tracing. This option is best used when accounting data is required only for a few applications in the host system. If NTSACCT=ALL is specified, then NTS starts tracing all network activity and collects accounting information for all sessions regardless of any *Saw class* parameters. If NTSACCT=NO is specified, then NTS does not collect any accounting information regardless of session parameters. Should any accounting be active when this is requested it ceases immediately. Once session awareness processing is active, only NTSACCT=NO can be specified. To change the SYSPARMS accounting value to either of the other options while session awareness is active, it must first be stopped, the command entered, and then session awareness restarted.

Default: SELECTIVE

NTSCINTV=*nn*

Specifies the NTS correlation interval value, where *nn* is a number of seconds. This value represents the maximum interval that NTS is prepared to wait for certain different types of data to be correlated. For example, trace information can arrive ahead of the session notification for a particular session. If this occurs NTS queues such data and solicits the latest session information from VTAM. However, if the session data for the traced session does not arrive within the elapsed time of the correlation interval all uncorrelated data is purged. Similarly, if there is outstanding data (such as trace or RTM data) yet to arrive after a session has ended, NTS is prepared to wait for it for up to the correlation interval specified. If this interval expires then logging proceeds regardless of whether all outstanding data has arrived. The default should be adequate for most networks.

Default: 30

NTSCLOSE={ NO | YES }

Specifies whether any sessions, which remain active when NTS is closed due to system initialization, are to be considered ended for the purpose of output logging. NTSCLOSE=NO means no active sessions are logged when NTS closes. If NTSCLOSE=YES is specified then those sessions still active are queued for output and time-stamped with the shutdown time. Such sessions are logged (according to the log options in effect) and are flagged with a C in subsequent NTS session selection lists indicating the session did not end normally but was closed by NTS. Because your product region only waits for a short period of time (about 10 seconds) between shutdown notification and actual termination, this technique is only useful where a small number of sessions remain which need to be logged. If a large number of sessions require closing before shutting down, the CLOSE operand of the NTS SAW STOP command is used to similarly close and log all active sessions before stopping.

Default: NO

NTSCNMQ={ NO | YES }

Specifies whether NTS is to queue NTS CNM requests. NTSCNMQ=YES means that all CNM requests are queued until they are replied to by VTAM. NTSCNMQ=NO means that NTS CNM requests are not queued, but issued immediately. Issuing NTSCNMQ=NO while NTS is active results in all queued NTS CNM requests being purged.

Default: YES

NTSEVENT={ NO | YES }

Controls NTS event generation. If you want events to be generated then you must specify SYSPARMS NTSEVENT=YES. Issue SYSPARMS NTSEVENT=NO to immediately terminate all NTS event generation irrespective of what class definitions are in place. Subsequently issuing SYSPARMS NTSEVENT=YES restarts generation as per the current class definitions.

Default: NO

NTSINTSV={ NO | YES | *name_mask* }

Specifies whether NTS intensive message recording is active. By enabling this function, certain activities can be tracked, such as the purging of uncorrelated trace data, and dumping of unrecognized entries in the trace buffer. This option is not usually required under normal operation.

name_mask

Specifies a mask that enables intensive message recording for certain LUs and PUs only.

Wildcards: * (any number of characters) and ? (one character)

Default: NO

NTSMAIEX={ NO | YES }

Specifies whether MAI sessions are presented to the NTS user exit. NTSMAIEX=NO indicates that NTS will not present MAI sessions to the NTS user exit. NTSMAIEX=YES indicates that an SMF type 39 record will be built for MAI sessions and be presented to the NTS user exit.

Note: For information about the record format, see the CA NetMaster NM for SNA *Administration Guide*.

Default: NO

NTSMAISV={ NO | YES }

Requests that MAI inform NTS of any current MAI session and subsequent MAI sessions. NTS must be activated before MAI session visibility becomes effective. SYSPARMS NTSMAISV=YES can be specified after NTS and MAI are already active and causes MAI to notify NTS of all currently existing MAI sessions. If SYSPARMS NTSMAISV=NO is specified after the interface has been active, any MAI sessions currently visible to NTS remain so but MAI does not notify NTS of any new MAI sessions.

Default: NO

NTSMAXTP=*nnn*

Provides the maximum number of trace PIUs that can be specified by any class definition, or operator command, as either the initial or final trace queue depth for any session.

Default: 100

NTSMAXTR =*nnn*

Provides the maximum number of resources that can have a specific STRACE request outstanding. This includes both trace start requests in normal operation, and trace stop requests where global tracing is active and the resource is to be excluded. It also includes those resource for which tracing was started by NTS to collect accounting data.

Default: 100

NTSRSINT=*nnn*

Specifies the interval length for resource statistics collection in minutes.

Default: 30

Limits: 1 to 480 (eight hours)

NTSRSLIM=*nnn*

Specifies how many of the above intervals can occur before the statistics collected for the oldest interval are overwritten. This operand can be overridden selectively using the LIMIT operand of the DEFCLASS RESOURCE command. If this parameter is not specified and the relevant NTS resource class does not specify a LIMIT parameter, the default number of intervals is used.

Default: 16

Limits: 0 to 255

NTSRSTAT={ NO | YES }

Specifies whether resource statistics are to be collected. If this parameter is not specified, or NTSRSTAT=NO is specified, then resource statistics collection is disabled. This means that no resource statistics will be collected, irrespective of options set in resource class definitions. Once session awareness (or SAW) is enabled, only NTSRSTAT=NO can be specified.

Default: NO

NTSSAWBF=(*n,s*K)

Specifies the number *n* (range 2 to 255) and size *s* (range 2 to 32) in KB of the buffers to be allocated by VTAM for collecting session awareness data. If not specified a default of 2 buffers of size 4 KB will be used.

NTSSKEEP=*nn*

Specifies the system default session keep count for sessions written to the NTS database. The number *nn* represents the maximum number of session incidences kept in the database for any specific session name pair. When a session ends and is written to the database this value is placed in the session master record. It can be modified subsequently by the NTSDBMOD command. A session name pair is determined by the real name and network of both session partners. The default value of 10 means that only the 10 most recent sessions for a session name pair will be kept in the database. As each session ends and is written out it is added to the database until the session count is reached after which any new sessions replace the oldest ones.

Default: 10

Limits: 1 to 255

NTSSMFTM={ GMT | LOCAL }

All application timestamps records in type 39 SMF records consist of the first four bytes of the system TOD clock value, plus a 4-byte signed number for the time zone adjustment value, in seconds. By definition, the first four bytes represent GMT time in approximately 1-second intervals. However, NetView (NLDM) writes the first four bytes of these timestamps in local time.

- To cause NetMaster to write timestamps in type 39 SMF records in local time, specify NTSSMFTM=LOCAL.
- To cause NetMaster to write timestamps in type 39 SMF records in GMT time, specify NTSSMFTM=GMT.

Default: GMT

NTSTRBFX={ NO | YES }

Specifies whether the trace final queue buffers are to be consolidated when the first wrap occurs. For sessions experiencing heavy traffic, the trace final queue might wrap quite frequently. If trace buffers are allocated from a large set of pages, and tracing is enabled for a large number of sessions, the paging overhead on the system might be considerable. Specifying NTSTRBFX=YES causes NTS to use a contiguous buffer pool for the allocation of trace buffers for a single session. This effectively fixes the pages in memory and therefore reduces the paging overhead. The movement of the contents of the trace buffers on the trace final queue into the contiguous storage buffer occurs when the trace final queue wraps for the first time.

Default: YES

NTSTRCBF=(*n*,*s*K)

Specifies the number *n* (range 2 to 255) and size *s* (range 2 to 32) in K of the buffers to be allocated by VTAM for collecting trace data. If not specified a default of 4 buffers of size 4K is used.

OCSHLITE={ NONE | REVERSE | BLINK | USCORE }

Specifies the type of highlighting to be used for messages in OCS windows normally displayed in high intensity. The following options can be specified:

NONE specifies that messages are presented without change.

REVERSE specifies that messages are presented in reverse video.

BLINK specifies that messages are to blink.

USCORE specifies that messages are underscored.

For terminals that support color, the default color applies. This operand is ignored for terminals that do not support IBM extended highlighting and does not apply to individual high intensity fields resulting from comment lines from NCL procedures that commence with a plus sign (+) and that use the at symbol (@) field highlighter.

Default: NONE

OCSTIME={ NO | YES }

Specifies whether the time appears at the end of the title line of an OCS window. If YES is specified, the current time in the format HH.MM is placed at the left hand end of the title line of an OCS window each time the window display is updated. This allows operators to determine when the last message occurred if the terminal has been left temporarily unattended. Specifying NO resets this option.

Default: YES

PANLBFSZ=*number*

Specifies the maximum outbound data stream size (in KB) that can be generated for any terminal attached to your product region.

If your network contains terminals that might receive large data streams (for example, complex extended data stream screen formats) you might need to increase the PANLBFSZ operand. A message is issued if an attempt is made to display a panel that is too large for the current PANLBFSZ setting.

Default: 16

Limits: 4 to 20

PANLBUFF=*number*

Specifies the maximum number of pages of virtual storage to be used for concurrent terminal output operations. If 0 is specified, no limit is imposed.

This parameter acts as a throttle for simultaneous output to large numbers of terminals (for example, during broadcast processing or when starting a large EASINET network). Increase this number to speed up network start-up and broadcasting, decrease it to throttle back these activities.

Before increasing this value, consider any effects an increase in virtual storage usage might have on other functions.

Default: 40

Limits: 0 to 32767

PPOCOLOR=*color*

Specifies the color to be used when displaying unsolicited VTAM (PPO) messages on OCS consoles.

PPO messages are displayed in high intensity on non-color devices.

Default: WHITE

Limits: BLUE, RED, PINK, GREEN, TURQUOISE, YELLOW, WHITE, and NONE

PPOHLITE={ NONE | REVERSE | BLINK | USCORE }

Specifies the type of extended highlighting used for unsolicited VTAM messages sent to OCS windows. The following options can be specified:

NONE Messages are presented without change.

REVERSE Messages are presented in reverse video.

BLINK Messages are to blink.

USCORE Messages are underscored.

This operand is ignored for terminals that do not support IBM extended highlighting.

Default: NONE

PPOSOMSG={ LOG | NOLOG | LOGDEFMSG }

Specifies how PPO messages, received from VTAM as the result of VTAM commands entered from a system console or a local OCS window, are to be logged.

VTAM relays copies of the results of commands entered through system consoles if the MODIFY PPOLOG=YES VTAM command has been entered, or if PPOLOG=YES is specified in the VTAM initialization parameters.

The following options can be specified:

LOG

All PPO messages are logged. This is the default.

NOLOG

PPO messages are ignored.

LOGDEFMSG

Only DEFMSGed PPO messages are to be logged (that is, those which have one or more of the three delivery options enabled or EDS delivery enabled).

PPOUSMSG={ LOG | NOLOG | LOGDEFMSG }

Specifies how unsolicited PPO messages are to be logged, regardless of DEFMSG options. The following options can be specified:

LOG

All unsolicited PPO messages are logged. Further delivery of unsolicited messages is still performed according to DEFMSG rules.

NOLOG

No logging of unsolicited messages. This is the default.

LOGDEFMSG

Only DEFMSGed unsolicited messages are logged (that is, those which have one or more of the three delivery options enabled or EDS delivery enabled).

PWEXPIRE=*number*

Specifies the number of days after which users are forced to change their password. This operand takes effect the next time a user logs on. This operand has no effect if a security exit is in force that replaces the UAMS password maintenance functions. Enter a value of 0 to disable automatic password expiry.

Default: 30

Limits: 1 to 366

PWMAX=*number*

Specifies the maximum acceptable length for passwords. This operand only takes effect from the next password change a user makes. The operand has no effect if a security exit is in force that replaces the UAMS password maintenance functions.

Default: 8

Limits: Value of PWMIN to 8

PWMIN=*number*

Specifies the minimum acceptable length for passwords. This operand only takes effect from the next password change a user makes. The operand has no effect if a security exit is in force that replaces the UAMS password maintenance functions.

Default: 3

Limits: 1 to the value of PWMAX

PWRETRY=*number*

Specifies the number of password violations that are accepted before a logon attempt is denied. If the number is reached, a warning message is sent to all terminals with Monitor status, advising them of the user ID and terminal involved in the violation. This operand has no effect if a security exit is in force that replaces the UAMS password maintenance functions.

Default: 2

Limits: 1 to 10

ROUTCDE=(*n,n,n*)

Specifies the operating system routing codes to be used for unsolicited messages sent to the system console (that is, to the *pppp*OPER user ID).

Note: For an explanation of the impact of specifying certain codes, see the appropriate operating system manual.

Default: 1,8,11

Limits: 1 to 16

RXCSTACK=*n*K

Specifies the working stack size (in KB) of the REXX compiler.

Default: 3

Limits: 3 to 16

RXISTG=*n*K

Specifies the size (in KB) of the initial storage to get for a REXX process. A nonzero value avoids startup overhead.

Default: 32

Limits: 0 to 128

RXMCSZ=*n*K

Limits the maximum size (in KB) of a compiled REXX object.

Default: 256

Limits: 64 to 512

RXMSTG=*n*K

Specifies the maximum amount of storage (in KB) a REXX process can use.

Default: 1024

Limits: 10 to 1024

RXQSFIX={ NO | YES }

IBM REXX interpreters let quoted strings extend across multiple lines. The GREXX Compiler does not handle this by default. If RXQSFIX is set to YES, then quoted strings that extend across multiple lines are converted with all trailing blanks stripped. The string still cannot be larger than 250 bytes in length.

Note: There is one circumstance where the resulting REXX source, while syntactically correct, may not execute correctly. If the repaired string has a REXX operator of higher precedence than concatenate (||), the resulting expression is processed in the wrong order.

Default: NO

RXRETAIN=*n*K

Specifies the size (in KB) of the REXX retained procedure pool.

0 specifies that no dynamically loaded procedures are retained.

Default: 200

Limits: 0 and 10 to 2048

RXXSTACK=*n*K

Specifies the working stack size (in KB) of a REXX process.

Default: 4

Limits: 3 to 16

SESSMSG={ NO | YES }

Specifies whether trace message N35007 is issued each time a session to a remote system opens or fails. If YES is specified, the system issues the trace message to Monitor status users, each time a session to a remote system is opened or fails. The system retries any failures indefinitely at intervals specified in the LINK command until the session is opened or the link is stopped.

This option can be particularly useful in an INMC operation, as it might be the only way to identify the fact that not all sessions making up a link are operational. If any sessions are open the link remains operational-the failure of an individual session does not disrupt traffic, but can affect performance. The default is NO.

Default: NO

SMFTRACE=*number*

Requests the dump of written SMF records.

STGWRN=*number*

Specifies the number of kilobytes below the 16-MB line at which an N01801 message is issued as a WTO indicating that the storage thresholds have been exceeded. This message can be repeated at 30-second intervals until the storage use drops below the threshold.

Default: 0 (no warning limit)

Limits: 0 to 16,000

STGWRNXA=*number*

Specifies the number of kilobytes above the 16-MB line at which an N01801 message is issued as a WTO indicating that the storage thresholds have been exceeded. This message can be repeated at 30-second intervals until the storage use drops below the threshold.

Default: 0 (no warning limit)

Limits: 0 to 1,000,000

SYSCONMU=*name*

Specifies the default user ID for a master console that issues commands to your region when it is not signed on to the security system in use.

SYSCONMU can be set only during NMINIT. For master consoles that are not signed on to your security system, this user ID always applies, regardless of the setting of the SYSPARM SYSCONSO.

If this default user ID is not defined to your security system, then the user ID .MASTOP (which has a hard-coded profile) is used.

Note: For more information about master console user ID requirements, see the *Security Guide*.

Default: *pppp*MSOP, where *pppp* is the system user prefix

Limits: One to eight characters

SYSCONNM={ EXTMCS | ALL }

Specifies the LU name that is assigned to system console environments. The following options can be specified:

- EXTMCS specifies consoles with console IDs in the range 0 to 99. The consoles are named CONSOLE (0) or CONS#*nn*, where *nn* is 01 to 99. Console IDs outside this range (that is, extended MCS consoles) use the extended MCS console name.
- ALL specifies that the LU name is always the z/OS-assigned console name. This means that Console 0 is INTERNAL, the master is MASTER, and so on.

Default: EXTMCS

SYSCONSO={ DEFAULT | NO | REQUIRED }

Specifies whether the console user ID can default and whether signon is required. It is relevant to all operating system environments. The following options can be specified:

- DEFAULT specifies that the default SYSCONUI user ID name is used to sign on if the system console user ID is not defined in the UAMS security system.
- NO specifies that all consoles are signed on using the SYSCONUI user ID name.
- REQUIRED specifies that the user ID must be defined, otherwise the signon fails. This option is typically only used if SYSCONXU=YES is in effect.

Note: There are interactions between the SYSCONSO and SYSCONXU parameters.

Default: DEFAULT

SYSCONUI=*name*

Specifies the default system console signon name. The *name* must be a valid user ID.

Note: The user ID is not used unless it exists in the security system in use.

Note: For more information about console user ID requirements, see the *Security Guide*.

Limits: One to eight characters

SYSCONXU={ NO | YES }

Specifies whether or not external console user IDs are to be used when signing on consoles. The following options can be specified:

- NO specifies that the system-supplied user ID is not used.
- YES specifies that the system-supplied user ID is used. However, it is only used if the system passes a RACF UTOKEN with the command. If no token is passed, or the console is not signed on to RACF, the console is treated as not signed on, and the action taken depends on the value of SYSCONSO. If the master console user ID, *BYPASS*, is seen, a special internal user ID of .MASTOP is used to indicate that this is the master console. It is not signed on to RACF.

Note: There are interactions between the SYSCONXU and SYSCONSO parameters.

Default: NO

SYSLOG={ NO | YES | PPO }

Specifies whether none, all, or unsolicited VTAM messages written to the activity log, are also written to the system log. The following options can be specified:

- NO writes no messages to the system log.
- YES copies everything written to the activity log to the system log.
- PPO writes all unsolicited VTAM messages to the system log and the activity log.

Default: NO

SYSLOGFM={ MVS | MSP }

Specifies the format of the SYSLOG lines produced if SYSLOG=YES or SYSLOG=PPO.

Format a line with four zeroes for ROUTCDE, time, and user ID in the JOBID column.

TNDSREG={ NO | YES }

Specifies whether the Telnet Server will register new connections with the Packet Analyzer, for use by any CA NetMaster NM for TCP/IP regions on the same z/OS image.

Setting NO means connections are not registered.

Default: YES

TRACEOPT=*cc*

Specifies the trace options to be applied when tracing data streams sent to or from a terminal. The value of *cc* is the character representation of a hexadecimal byte.

The following bit values represent valid trace options:

```
B'10000000'(X'80')   Trace only first 256 bytes of each
                     message.
B'00000001'(X'01')   Trace output before compression.
B'00000010'(X'02')   Trace output after compression.
B'00000100'(X'04')   Trace input from terminal.
```

Default: 06

Example:

```
SYSPARMS TRACEOPT=80
```

indicates that the tracing options required correspond to a hex byte with a value of X'80'. This byte in turn represents an 8-bit string with the value:

```
B'10000000'
```

Example:

The value specified on the TRACEOPT parameter can be any combination of the four options, expressed as a hex character, for example:

```
SYSPARMS TRACEOPT=84
```

indicates a request to trace the first 256 bytes received from the terminal, with the data stream being written to the activity log. Data recorded can then be examined using the standard online log browse facilities.

Tracing is started and stopped by the LUTRACE command.

USERPW={ NO | YES | VERIFY }

Specifies whether the NCL system variable &USERPW is available for use in MAI logon data. &USERPW represents the user's product region password and is used when MAI sessions are created. This operand allows installations to control the availability of the &USERPW variable to MAI.

If YES or VERIFY is specified, the password is encrypted in storage so that it is not available in plain text. Specifying VERIFY indicates that MAI should prompt users for their password when a session is updated or added that contains &USERPW in the logon data.

Default: VERIFY

VDISPLAY={ CMD | MSG | ANY }

Specifies how the VTAM display command (D) is processed for users with command Network Partitioning. The following options can be specified:

- CMD lets users display only those resources within their defined command partitions.
- MSG lets users display only those resources within their defined message partitions. This can be used where resources within command and message partitions differ. In this case, the Inactive or Active status of message tables is ignored and all message tables are searched.
- ANY lets users display any resource, regardless of the resources specified within their partitions.

Default: CMD

VTAMID=*procname*

Specifies the system procedure name used for starting VTAM. If any other procedure name is used, then your product region must be informed of it so that the correct name can be used when generating VTAM commands.

Default: (ACF/VTAM and VTAM-G) NET or (ECS/VTAM) VTM

Limits: One to eight characters

XABELOW={ NO | YES }

Specifies whether your product region is to allocate buffer storage below the 16-MB line in XA systems if all storage in the extended private area has been used. NO means that if all XA storage in the product region address space has been used, further requests for XA storage by other processes fails, even if non-XA storage is still available below the line. Do not change this unless your installation requires your product region to run with a severely limited extended private area.

Default: NO

Appendix D: MAI Installation Exit MAIEX02

This section contains the following topics:

MAI Installation Exit (see page 295)
Implement the MAI Exit (see page 296)
Start the Exit (see page 298)
Registers on Entry to the Exit (see page 299)
Exit Correlators (see page 299)
Contents of the Communications Area (see page 300)
Return Codes from the Exit (see page 304)

MAI Installation Exit

The installation-written exit MAIEX02 is supported by your product region. It can be invoked by MAI for exit initialization and termination and for MAI session initiation and termination. The exit can be used to perform the following functions:

- Validate the various parameters to be used on a session
- Refuse the session or change the parameters to be used
- Collect session accounting information
- Supply an LU name or LU name prefix to be used for a session
- Limit the number of sessions

Implement the MAI Exit

The exit must be link edited and placed into a load library accessible to your product region. In z/OS and z/VM systems, the exit must be link edited with the RENT option.

The name of the link edited exit module is identified to your product region by the SYSPARMS MAIEX02 operand.

The exit is normally written in assembler and can perform any processing required, including SVC calls and WAIT macros. This is because it operates under a subtask that is independent of the main product region task.

Note: Although executed under a subtask, the termination call to the exit when your system closes down is serialized and the termination call must complete before termination processing resumes.

Reentrant Code

You must write the exit to conform to the following rules:

- It is written in reentrant code
- It does not modify any storage location within itself
- Any working storage required other than that provided in the communications area is obtained using GETMAIN.

Failure to observe these conditions will result in an abend in the processing task and the session request will fail. Other processing is unaffected.

Storage Subpools

It is possible for the exit to maintain information across calls in GETMAIN storage, remembering the addresses of the storage in the various correlators. If this technique is to be utilized, the storage must be obtained in subpool 50. Storage obtained in any other subpool may be automatically freed when the exit returns to MAI or, in the case of storage obtained on a Session Start call, when that MAI session ends.

It is the responsibility of the exit to free any GETMAIN storage when necessary. MAI will never free storage obtained by the exit.

Exit Serialization

Because MAIEX02 is run in a subtask, it is possible for it to run concurrently for two or more different users. For example, while it is processing a Session Start call for one user, it could be called to process a Session End call for another. This could lead to complications where, for instance, the exit is maintaining a control block structure or changing a correlator. This is because the subsequent call may interrupt processing in the first call.

MAI provides two levels of serialization to overcome these problems, governed by the SYSPARMS MAIEX02S command.

First Level Serialization

The first level of serialization uses MAIEX02S=SYSTEM. This is the default and ensures that calls to the exit are totally serialized. In the above example, the Session End call would not be made until the Session Start call had completed. This level is used where, for instance, the exit is maintaining a control block structure or where the various calls update the System Correlator. It ensures that a System Correlator set by one call is always passed to the next call.

The only drawback with this level of serialization is that the starting and ending of MAI sessions is *single-threaded* so that only one can proceed at a time. However, because the path through MAIEX02 is usually short and fast, this should not create a problem.

Second Level Serialization

The second level of serialization, MAIEX02S=USER, ensures that calls to the exit are serialized at a user level only. That is, the exit can concurrently process multiple calls for different users, but calls for any one user will be serialized. This level protects the User Correlator, ensuring that if set by one call it is presented to the next call for that user correctly. However, it does not protect the System Correlator or any control block structure the exit may maintain.

MAIEX02S=USER provides processing overlap advantages over the MAIEX02S=SYSTEM option and can be used with safety if the exit is purely validating or changing session parameters.

Sample Exit

A sample exit, MAIEX02, is installed with your product in CC2DSAMP. We recommend that you assemble and study this before writing an exit. The macro $NMMAEX2 in the CC2DMAC data set must be available for this assembly to function correctly.

The sample shows a way in which the correlators can be used to count the number of MAI sessions a user has and the total number at any one time. It then limits the number of sessions allowed and rejects sessions if the counts pass a predetermined level. Because Session Start and End calls to the sample exit update the System Correlator, it requires the MAIEX02S=SYSTEM serialization level.

This sample also includes support for generating Pass tickets as used by the Secured Signon function of RACF and other external security products.

Start the Exit

To start the MAIEX02, use the SYSPARMS MAIEX02 command, naming the program that is to act as the exit. If the SYSPARMS command is included in the RMINIT procedure, an Exit Initialization call is made. Subsequent MAI session requests then call the exit. If the exit is not included in RMINIT, but is invoked using command from a terminal, any existing users with MAI sessions may have to log off before any calls are made to the exit for that user.

During the testing phases of the exit, it may be necessary to force a new Initialization Call or to invoke an entirely different program. This can be accomplished by entering another SYSPARMS MAIEX02 command, specifying the same or a different program name. When the command is entered, an Initialization Call is made, with a zero System Correlator and any existing User Correlators are zeroed. In addition, MAI ensures that if any MAI sessions are currently in progress, the ending of those sessions does not cause a Session End call to be made to the old or the new exit.

Registers on Entry to the Exit

When the exit is invoked, Register 1 contains the address of a fullword, which in turn contains the address of a communications area containing various parameters. This communications area can be mapped using a supplied macro, called $NMMAEX2. This macro provides a DSECT expansion to perform the mapping and detailed information about the content of each field.

Standard linkage conventions apply. On entry, the exit must save the contents of all registers (Register 13 contains the address of a save area) and on exit all registers must be restored to their contents on entry, with the exception of Register 15 which should contain a return code.

Exit Correlators

MAIEX02 has available to it a number of correlator areas, each one a fullword in length. These correlator fields can be used for any purpose. For example, they can contain counts or storage addresses. The exit can supply or change the content of any correlator supplied during any call.

System Correlator

One System Correlator exists in a system. It is supplied in all calls to MAIEX02. If any call updates it, that updated value is returned in all subsequent calls.

User Correlator

One User Correlator exists for each logon session. That is, when a user first logs on, that user is allocated a correlator containing hexadecimal zeroes. If an MAI session start, ACB open, or session end call updates the User Correlator, that updated value is returned in all subsequent session start, ACB OPEN and session end calls.

Session Correlator

When a user starts an MAI session, that session is allocated a correlator containing hexadecimal zeroes. If a session start or ACB open call updates the Session Correlator, that updated value is returned in subsequent calls for that session.

Security Exit Correlator

Session start, ACB open and session end calls are provided with the security exit correlator. This correlator is a user-level correlator that can be provided by the security exit.

Note: For more information, see the *Security Guide*.

Security Exit User Token

Session start, ACB open and session end calls are provided with the security exit user token. This is a user-level token that can be provided by the security exit. For more information, see the *Security Guide*.

Contents of the Communications Area

The exit has the following types of call, and the content of the communications area depends on the type of call:

- Exit initialization
- Exit termination
- MAI session start
- ACB open
- MAI session end

Exit Initialization Call

The Exit Initialization call is made when a SYSPARMS MAIEX02 command is entered to invoke the exit. The following parameters are contained in the communications area:

- Function code, F'0'
- Address of the communications area
- ID of this system
- System Correlator
- A work area to be used for any purpose

Until the exit returns from an Initialization call, no MAI sessions are allowed.

Exit Termination Call

The Exit Termination call is made when your product region is about to terminate. The following parameters are contained in the communications area:

- Function code, F'4'
- Address of the communications area
- ID of this system
- System Correlator
- A work area to be used for any purpose

MAI Session Start Call

The MAI Session Start call is made when a new MAI session is about to start. The following parameters are contained in the communications area:

- Function code, F'8'
- Address of the communications area
- ID of this system
- Security Exit correlator
- User ID of user starting the session
- Node name of terminal from which session is being started
- Session ID of session being started
- Terminal's screen sizes
- Flags to indicate whether the session is MAI-OC
- Language code of the user
- System Correlator

Items from this point onwards can be modified by the exit and the modified information will be used when the session is created:

- User Correlator
- User Security Token
- Session Correlator
- Any LU name chosen by the user
- MAI node name prefix as designated by a SYSPARMS MAIOPREF command
- The application with which the session will be started
- The LOGMODE name chosen by the user or MAI
- The length of any user data to be passed to the application at LOGON
- The user data itself
- An area into which the exit may place an error message if the session is to be refused
- A work area to be used for any purpose
- An indicator word, into which the exit places a value of F'4' if it requires an ACB open call
- The name of an NCL procedure to perform session script functions, and optional parameters to be passed to it

ACB Open Call

The ACB Open call is made after MAI has successfully opened or allocated the ACB to be used for the session. However, the call is only made if the exit returned the appropriate value in the indicator word after the Session Start call. The following parameters are contained in the communications area:

- Function code, F'16'
- Address of the communications area
- ID of this system
- Security Exit correlator
- User ID of user starting the session
- Node name of terminal from which session is being started
- Session ID of session being started
- Flags to indicate whether the session is MAI-OC
- The MAI-FS privilege class of the user
- System Correlator
- User Correlator
- User Security Token
- Session Correlator
- Name of the ACB which will be used on this session
- Name of the application with which the session will be started
- A work area to be used for any purpose
- The length of any user data to be passed to the application at LOGON
- The user data

MAI Session End Call

The MAI Session End call is made when an MAI session has ended. The following parameters are contained in the communications area:

- Function code, F'12'
- Address of the communications area
- ID of this system
- Security Exit correlator
- User ID of user on whose behalf session was started
- Node name of terminal from which session was started
- Application with which session was started
- Session ID of session ending
- Flags to indicate whether session is MAI-OC and whether any WAIT=PERM specification for the session is to be canceled
- System Correlator
- User Correlator
- User Security Token
- Session Correlator
- A work area to be used for any purpose
- The count of inbound and outbound bytes over the duration of the session
- The count of inbound and outbound RUs over the duration of the session

Return Codes from the Exit

When the exit returns to MAI-FS, Register 15 should contain a return code for the Exit Initialization call and the Session Start call.

Exit Initialization

Register 15 should contain zero if exit initialization was successful. A nonzero value in Register 15 indicates that initialization was not successful. In this case, no MAI sessions are allowed. Any attempts to start an MAI session are rejected with an appropriate error message.

Session Start

Register 15 should contain zero if the session should proceed using the parameters in the communications area. Any value other than zero indicates that the session will be refused and the message placed in the message field of the communications area displayed at the user's terminal. If this field is blank, the following default message is displayed:

```
SESSION REFUSED BY INSTALLATION EXIT
```

Appendix E: Consoles and Migration ID Exits

This section contains the following topics:

Console Use (see page 307)
JES, OP1, OP2, and Pseudo Consoles (see page 307)
Extended MCS Consoles (see page 308)
Console Management (see page 309)
Specify the Number of Consoles (see page 310)
Use Migration ID Exits to Customize Console Management (see page 311)
User Exits for Migration IDs (see page 312)

Console Use

For operators to be able to issue system commands from an Automation Services region, it is essential that you define suitable consoles to the system before starting the region. An Automation Services region can use JES or extended multiple console support (MCS) consoles in a z/OS environment, OP1 or OP2 consoles in an MSP environment, and pseudo consoles in a VOS3 environment.

JES, OP1, OP2, and Pseudo Consoles

JES, OP1, OP2, and pseudo consoles are virtual consoles. They can be acquired by any authorized program for use in issuing system and subsystem commands.

Extended MCS Consoles

In the z/OS environments, Automation Services can use extended MCS virtual consoles.

The advantages in using extended MCS consoles are:

- There is no theoretical limit to the number of extended MCS consoles in an z/OS configuration. JES consoles are limited to 99 across a sysplex.
- You can have a MASTER authority level.
- In a sysplex configuration, extended MCS consoles are more flexible.

Automation Services uses extended MCS consoles as follows:

- The eight-character name of consoles is constructed by using the convention specified in the CONSOLES parameter group, which is accessible through the /PARMS panel shortcut. The extended MCS console prefix is further prefixed with a Z and padded to five characters with Zs. The last three characters are a decimal number from 001 through 255. The region can use up to 254 extended MCS consoles.
- Console authority of MASTER can be set and honored.

Migration IDs

It is possible to assign a unique migration ID to an extended MCS console. The ID is required to issue commands to those applications that do not support named consoles. Across a sysplex, there is a limit of 150 migration IDs.

When a region issues commands internally by using an extended MCS console, it decides whether a migration ID is required through its migration ID determination exit. The exit requests an ID for all MODIFY and STOP system commands, and for any unrecognized commands.

When users issue commands in the region by using the SYSCMD command, they can choose whether a migration ID is required by using the MIGID operand of the SYSCMD command. If the MIGID operand is not specified, the setting specified in the CONSOLES parameter group is used. The default setting is YES, specifying that IDs be used.

The limit of 150 migration IDs may be a problem in a large sysplex where many applications are using extended MCS consoles. If a region cannot acquire a console when requested, the requesting SYSCMD command fails. Use the following suggestions to help you correct the problem:

- In the CONSOLES parameter group, change the value in the Acquire with Migration ID (default) field to NO.
- Consider writing your own migration ID determination exit to further restrict the use of IDs by internally issued commands. A sample exit, NMMIGIDX, is supplied with the product.

Console Management

A region uses the CONSOLES parameter group to specify its console requirements.

Automation Services uses consoles as follows:

- During region initialization, the CONSOLES parameter group sets a limit on the number of consoles available to authorized users in the region.
- Consoles are acquired and released as necessary.

Console Pool Management

The CONSOLES parameter group contains the following two values that govern how the region manages its pool of consoles:

- Maximum number of consoles that can be acquired concurrently by the region
- Maximum number of free consoles that should be retained in a pool for the region when the region finishes using them

Console Acquisition

When a user issues the SYSCMD command, the region tries to acquire a console as follows:

- It uses a console previously assigned to the user if the console has not timed out and if the console attributes match the new request.
- It searches the pool of consoles for a free console. If one is found, it is temporarily assigned to the environment that is issuing the command.
- If there are no free consoles and if the number of consoles currently acquired is less than the specified maximum, it tries to acquire one from the system.
- If the maximum number of consoles is already acquired, the acquisition fails. For commands issued internally, the region retries the acquisition as specified by the CONSOLES parameter group.

Console Release

When a console is not used any more, it is put in the pool. When the number of free consoles exceeds the value specified in the CONSOLES parameter group, the extra consoles are released back to the system.

Specify the Number of Consoles

Different products have different requirements for consoles. Some products may have a peak in the number of consoles required at region startup. You should allocate the appropriate number of consoles to handle the consumption peaks.

Note: Because more extended MCS consoles are available than JES consoles, use extended MCS consoles in preference to JES consoles when it is feasible.

Use the SHOW CONSOLES command to review console usage by the region.

Considerations

If a region requires a large number of consoles, then the larger the number of consoles retained, the more efficient the region becomes. However, retained consoles are not available to other applications.

Consider the following when specifying the Max Consoles to Retain value in the CONSOLES parameter group:

- The number of MSP OP1 and OP2 consoles is limited to 10 per system.
- The number of z/OS JES consoles is limited to 99 across a sysplex.
- It might be necessary to limit the number of consoles to retain if a significant number of extended MCS consoles require migration IDs.

Use Migration ID Exits to Customize Console Management

When you are using Extended MCS (EXTMCS) consoles, not all software can handle a named console; therefore, regions can request a migration ID (as a pseudo-console ID) for an EXTMCS console. Because migration IDs are limited to a pool of 150 across a sysplex, you need to be able to determine the need for a migration ID on a case-by-case basis.

Note: You can use the CONSOLES parameter group in Customizer to set the various console parameters.

Allow Dynamic Determination of the Need for Migration IDs

You can allow dynamic determination of the need for a migration ID as follows:

- By issuing a SYSCMD command with the MIGID operand set to MIGID=EXIT.

  Specifying this operand, with some command text, causes your region to invoke its internal migration ID determination exit and, optionally, a user-written exit (that can override the internal exit decision).

- By using the NCL &AOMMIGID built-in function, in the format:

  ```
  &result = &AOMMIGID command-string
  ```

  The return value from the function is YES or NO, indicating whether the command needs a migration ID.

  For more information about the &AOMMIGID built-in function, see the *Network Control Language Reference Guide*.

Note: If the current console type does not support or need migration IDs, the exit is not called and the system always acts as if NO is specified. This also includes the case where no consoles are acquired.

User Exits for Migration IDs

You can optionally define a user migration ID exit, which can examine and optionally override the decision made by the default exit.

Why Define a User Exit?

The default migration ID determination exit should normally be adequate. However, uses for a user exit include:

- Handling in-house subsystems that do not use a command recognition character (that can be set in the CONSOLES parameter group). This can involve parsing the command text.
- The migration ID determination exit sets the MIGID for unrecognized commands to YES. However, if the subsystem can handle named consoles, then the user exit can set NO for these subsystem commands.
- Overriding the standard settings for some z/OS commands.

Sample Exit

A sample exit, NMMIGIDX, is distributed in source form. It illustrates the coding required to produce a useful exit.

Activate the User Exit

To activate the user exit, specify an exit name in the CONSOLES parameter group in Customizer (enter **/PARMS**).

User Exit Requirements

The migration ID determination user exit must meet the following requirements:

- The exit must be assembled and link-edited into an accessible load library. Any name is suitable (the name is specified in the SYSPARMS command to activate the user exit).
- Because your product region executes in APF-authorized mode, the exit must be in an APF-authorized library.
- The exit must obey standard linkage conventions:
  - R1 contains the address of the parameter list.
  - R2 to R12 must be saved on entry and restored on exit.
  - R13 points to a standard save area.
  - R14 holds the return address and AMODE.
  - R15 holds the entry point address.
- The exit can be AMODE 24 or 31, and RMODE 24 or ANY. All parameters are passed below the 16M line. The exit is called in the link-edited AMODE. It returns using BSM 0,14 (with the value in R14 as at entry). However, BR 14 also works.
- The exit is called from the main task. It must not issue any operating system waits or use services that take a significant length of time. Doing so would severely impact processing.
- If the exit abends, it is disabled.

Input Parameters for User Exits

The migration ID exit is called with a parameter list.

Note: Although the exit is called in whatever AMODE it is linked with, all parameters are placed below 16 MB.

R1 points to a list of fullword addresses. These addresses in turn point to parameters. The format is as follows:

```
R1  ==> A(PARM1)
        A(PARM2)
        A(PARM3)
        A(PARM4)
        A(PARM5)
        A(PARM6+X'80000000')   (end of list)
```

PARM1

Function code. This is a fullword (DS F). The only value currently assigned is 0, meaning analyze the passed command.

PARM2

Four flag bytes. These are defined as follows:

Note: Only defined bits are shown here.

```
CMDFLAGS  DS   0XL4    FLAG BYTES
CMDFLAG1  DS   X       FLAG BYTE 1
CMD1RTPF  EQU  X'80'   1 - ROUTE CMD PREFIX DET'D
*                      (SKIPPED, REST EVALUATED)
CMD1MIGY  EQU  X'40'   1 - SOLVE SAYS NEED MIGID(0-NO)
CMDFLAG2  DS   X       FLAG BYTE 2
CMD2JES   EQU  X'80'   1 - APPARENT JES  CMD
CMD2SUBC  EQU  X'40'   1 - SUBSYS CMD (IN AOMSUBCM L)
CMD2MVSC  EQU  X'20'   1 - MVS CMD (IE FOUND IN TAB)
CMDFLAG3  DS   X       FLAG BYTE 3 (CMD INDS)
CMD3MDFY  EQU  X'80'   1 - MODIFY CMD
CMD3STOP  EQU  X'40'   1 - STOP   CMD
CMD3RPLY  EQU  X'20'   1 - REPLY  CMD (EXPL R ...)
CMD3SRPY  EQU  X'10'   1 - SHORT REPLY (#...)
CMDFLAG4  DS   X       FLAG BYTE 4
```

PARM3

Command buffer. This is a 128-byte buffer containing the command string, padded with blanks. There are no leading blanks, and at least one non-blank character. This is the complete command (uppercase).

PARM4

Length of the command. This is a fullword (DS F). It contains the non-blank length of the command text passed in PARM3. The value ranges from 1 to 126.

PARM5

Actual command buffer. This is the buffer (actually part of the buffer in PARM3) that contains the analyzed command.

Unless CMDFLAG1/CMD1RTPF is set, this is the same buffer (address) as PARM3.

If the route prefix flag is set, then this is the buffer (address) containing the routed command.

PARM6

Length of the actual command. This is a fullword (DS F). It contains the non-blank length of the command text passed in PARM5. The value ranges from 1 to 126.

Unless CMDFLAG1/CMD1RTPF is set, this is the same value as in PARM3.

If the route prefix flag (CMD1RTPF) is set, then this is the length of the routed command.

Note: Only one command buffer is used. If a ROUTE (RO) system command is detected, the second (actual) command buffer address is the address of the target command in the complete ROUTE command.

Return Codes from a User Migration ID Exit

A user migration ID exit informs your region of its decision by setting one of the following return codes (in R15):

0

No change to the MIGID setting that has been determined

4

Force MIGID=NO.

8

Force MIGID=YES.

No other return codes are defined, and any other return value is treated as RC=0.

Define Consoles to a CICS Region

In Automation Services, you specify the number of consoles for it to use in the Max Consoles to Acquire field of the CONSOLES parameter group. If you use Automation Services to manage a CICS region, you must define the required number of consoles to that region in the CICS Terminal Control Table (TCT) as follows:

For...	The required minimum is the...
JES consoles	number specified in the Max Consoles to Acquire field.
extended MCS consoles	number specified in the Max Consoles to Acquire field plus two.

Appendix F: Automation Services Application Program Interface

Note: This chapter applies to all products except CA SOLVE:FTS, and some products support only a subset of the APIs.

This section contains the following topics:

Application Programming Interface Procedures (see page 318)
$RMCALL ACTION=COMMAND (see page 324)
$RMCALL ACTION=DBGET (see page 329)
$RMCALL ACTION=PURGE (see page 331)
$RMCALL ACTION=STGET (see page 332)
$RMDBAPI SERVICE=CREATE (see page 334)
$RMDBAPI SERVICE=DELETE (see page 347)
$RMDBAPI SERVICE=GET (see page 349)
$RMDBAPI SERVICE=LIST (see page 352)
$RMDBAPI SERVICE=SET (see page 354)
$RMEVENT (see page 356)
$RMSTSET (see page 357)
$RECALL SERVICE=SET (see page 358)
$RECALL SERVICE=GET (see page 360)
$RECALL SERVICE=ACTION (see page 362)
$REDBAPI SERVICE=CREATE (see page 363)
$REDBAPI SERVICE=DELETE (see page 370)
$REDBAPI SERVICE=GET (see page 372)
$REDBAPI SERVICE=LIST (see page 374)

Application Programming Interface Procedures

The application program interface enables sources external to products that use Automation Services to call Automation Services functions, and retrieve information about resources and services.

The following API procedures are supplied for this purpose:

$RMCALL

Allows you to execute Automation Services commands, retrieve information about definitions in the knowledge base, retrieve resource and service status information, and delete extraneous link records.

$RMDBAPI

Allows you to maintain the ResourceView definitions and, in a CA NetMaster FTM region, file transfer rule sets and rules in the knowledge base.

$RMEVENT

Allows you to send a message to a (typically user-defined) resource or service.

$RMSTSET

Allows you to set the actual state of a defined resource or service.

$RECALL

Allows you to process EventView variables from NCL procedures.

$REDBAPI

Allows you to maintain the EventView definitions in the knowledge base.

The API can be called from:

- OCS or the system console
- Batch programs
- NCL procedures
- State change exits

Each of the APIs and their calling conventions are described in the following sections.

Note: *Services* are considered to be *a specific type of resource*—that is, a resource with a class of SVC.

$RMCALL API

$RMCALL is the main API procedure used to call Automation Services from external sources.

$RMCALL General Syntax

The general syntax for the $RMCALL API procedure is as follows:

```
$RMCALL         OPT=SERVICE
                SERVICE=ACTION
                ACTION=[COMMAND | DBGET | PURGE | STGET]
                [ACBNAME=acb-name]
                [NAME=resource-name]
                [CLASS=cc]
                [SYSNAME=system-name]
                [VERSION=version]
                [COMMAND=command-name]
                [SYNC={YES | NO | NOTIFY}]
                [NCLID=ncl-id]
                [PARMS={'parm1=value1 parm2=value2 parm3=value3 ...'}]
```

You must specify the OPT=SERVICE and SERVICE=ACTION operands as shown. The ACTION operand is used to specify the function that the procedure performs. The remaining operands may be optional depending on the function being performed. Operands for each function are discussed in the section that describes the function.

If you code more than one value for an operand (for example, the PARMS operand) you must separate parameters with spaces and enclose them in quotation marks.

More information:

$RMCALL ACTION=COMMAND (see page 324)
$RMCALL ACTION=DBGET (see page 329)
$RMCALL ACTION=PURGE (see page 331)
$RMCALL ACTION=STGET (see page 332)

$RMDBAPI API

$RMDBAPI is the API procedure used to maintain ResourceView definitions from external sources. It does not support:

- Shared and sysplex system image definitions
- Sysplex class resource definitions
- CA NetMaster NM for SNA and CA NetMaster NM for TCP/IP resource details

$RMDBAPI General Syntax

The general syntax for the $RMDBAPI API procedure is as follows:

```
$RMDBAPI        SERVICE={ACTIVATE|INACTIVATE|CREATE|DELETE|GET|LIST|SET}
                [TRUNCATE={YES|NO}]
                [{NAME=resource-name[MANNAME=manager-name]}|
                   {RSNAME=ft-ruleset-name[RMNAME=ft-rule-name]}]
                CLASS=cc
                [SYSNAME=system-name]
                [VERSION=version]
                [field-name-1=field value-1]
                [field-name-2=field value-2]
                .
                .
                .
                [field-name-n=field value-n]
```

The SERVICE operand is used to specify the function that the procedure performs. The remaining operands may be optional depending on the function being performed. Operands for each function are discussed in the section that describes the function.

More information:

$RMDBAPI SERVICE=CREATE (see page 334)
$RMDBAPI SERVICE=DELETE (see page 347)
$RMDBAPI SERVICE=GET (see page 349)
$RMDBAPI SERVICE=LIST (see page 352)
$RMDBAPI SERVICE=SET (see page 354)

$RMEVENT API

$RMEVENT is an API procedure used to send a message to a defined resource from external sources.

$RMEVENT General Syntax

The general syntax for the $RMEVENT API procedure is as follows:

```
$RMEVENT        CLASS={ USRCLS | class-name }
                NAME=resource-name
                MSG='message-text '
```

More information:

$RMEVENT (see page 356)

$RMSTSET API

$RMSTSET is an API procedure used to set the actual state of a defined resource from external sources.

$RMSTSET General Syntax

The general syntax for the $RMSTSET API procedure is as follows:

```
$RMSTSET        CLASS={USRCLS |class-name}
                NAME=resource-name
                STATUS=actual-state
```

More information:

$RMSTSET (see page 357)

$RECALL API

$RECALL is an API procedure used to maintain EventView variables and control rule sets from external sources.

$RECALL General Syntax

The general syntax for the $RECALL API procedure is as follows:

```
$RECALL     SERVICE={ACTION|GET|SET}
            [ACTION={ACT|INACT}]
            [CLASS=VARIABLE]
            NAME={'RULESET=rulesetname'|'VARNAME=variable-name'}
            [PARMS='VALUE=new-value']
            [DESC=value-description]
```

The SERVICE operand is used to specify the function that the procedure performs. The remaining operands may be optional depending on the function being performed. Operands for each function are discussed in the section that describes the function.

More information:

$RECALL SERVICE=SET (see page 358)
$RECALL SERVICE=GET (see page 360)
$RECALL SERVICE=ACTION (see page 362)

$REDBAPI API

$REDBAPI is the API procedure used to maintain EventView rule set and message rule definitions from external sources.

$REDBAPI General Syntax

The general syntax for the $REDBAPI API procedure is as follows:

```
$REDBAPI        SERVICE={CREATE|DELETE|GET|LIST}
                [TRUNCATE={YES|NO}]
                [NAME=rule-object-name]
                CLASS=cc
                [RULESET=ruleset-name]
                [RULEID=message-rule-id]
                [field-name-1=field value-1]
                [field-name-2=field value-2]
                .
                .
                .
                [field-name-n=field value-n]
```

The SERVICE operand is used to specify the function that the procedure performs. The remaining operands may be optional depending on the function being performed. Operands for each function are discussed in the section that describes the function.

More information:

$REDBAPI SERVICE=CREATE (see page 363)
$REDBAPI SERVICE=DELETE (see page 370)
$REDBAPI SERVICE=GET (see page 372)
$REDBAPI SERVICE=LIST (see page 374)

$RMCALL ACTION=COMMAND

Syntax

```
$RMCALL     OPT=SERVICE
            SERVICE=ACTION
            ACTION=COMMAND
            [NAME=resource-name]
            [CLASS=cc]
            [SYSNAME=system-name]
            [VERSION=version]
            COMMAND=command-name
            SYNC={YES | NO | NOTIFY}
            [NCLID=ncl-id]
            [PARMS={'parm1=value1 parm2=value2 parm3=value3...'}]
```

Use

Use this call to execute Automation Services commands.

Operands

Operand	Description
OPT=SERVICE	Indicates that an API service is to be performed.
SERVICE=ACTION	Indicates that an ACTION service is to be performed.
ACTION=COMMAND	Indicates that a command is to be processed.

Note: The following four operands (NAME, CLASS, SYSNAME, and VERSION) are included as operands in the API for the purpose of backward compatibility. Commands usually set these values, or prompt users for the values, depending on how the command is defined. If you do need to set these values (for example, you want to execute the command programmatically) set them through the PARMS operand. The values specified in the PARMS operand override the default values supplied by other operands.

Operand	Description
NAME=*resource-name*	Specifies the name of the resource to which the command applies.
CLASS=*cc*	Specifies the two-digit identifier of the ResourceView definition class (see page 383) to which the resource belongs.

Operand	Description	
SYSNAME=*system-name*	Specifies the name of the system image where the command is executed.	
VERSION=*version*	Specifies the version of the system image where the command is executed.	
COMMAND= *command-name*	Specifies the name of the command to be executed.	
SYNC={YES|NO|NOTIFY}	Specifies how the command is to be executed. The following values can be specified:	
	YES	The command will be executed synchronously: &RETCODE and &SYSMSG values are returned to the calling procedure.
	NO	The command is executed asynchronously: results are not returned to the caller.
	NOTIFY	The command is executed as for SYNC=NO but &SYSMSG values are returned to the caller's request queue in the form of $$$MSG$$$ *message-text,* where *message-text* is the contents of &SYSMSG.
NCLID=*ncl-id*	When SYNC=NOTIFY is specified you need to specify an NCLID so that the value of &SYSMSG can be returned to the appropriate request queue.	

Operand	Description
PARMS={`*parm1=value1 parm2=value2 parm3=value3 ...*'}	Use this operand to specify parameters for the command. Delimit the parameters with spaces. If you specify more than one parameter you need to enclose the parameter list in quotation marks. The following parameters can be set when executing a command (see the previous note): NAME=*resource-name* — The name of the resource to which the command applies. CLASS=*cc* — The two-digit identifier of the class to which the resource belongs. SYSNAME=*system-name* — The name of the system image where the command is executed. VERSION=*version* — The version of the system image where the command is executed. Some commands require other parameters (see page 327).

Returned Variable

Variable	Description
&SYSMSG	Contains the message returned by $RMCALL.

Return Codes

The following return codes indicate the success or failure of command processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Examples

The following example shows how to execute the CHECKALL command on the local system:

```
&CALL    PROC=$RMCALL +
         PARMS=(OPT=SERVICE,+
                SERVICE=ACTION,+
                ACTION=COMMAND,+
                COMMAND=CHECKALL)
```

The following example shows how to execute the ASA command against the printer (class 11) named RES001, located on version 0001 of the EASTPRD system:

```
&CALL    PROC=$RMCALL +
         PARMS=(OPT=SERVICE,+
                SERVICE=ACTION,+
                ACTION=COMMAND,+
                COMMAND=ASA,+
                PARMS='SYSNAME=EASTPRD VERSION=0001 CLASS=11 NAME=RES001')
```

Supplied Commands That Require Parameters

Two commands, GLOBAL and LOAD, require parameters when they are executed programmatically.

GLOBAL Command

Command Parameter	Description
MODE={MANUAL|AUTOMATED}	The GLOBAL command lets you specify whether the defined services and resources are to be controlled automatically or manually.

Note: If you specify AUTOMATED, resources and services that have MANUAL or IGNORED specified in their definitions are still controlled manually. If you specify MANUAL, then **all** elements are controlled manually.

LOAD Command

Command	Description
NEWSYS=*system-name*	Specifies the name of the new system image.
NEWVERS=*version*	Specifies the version of the new system image.
MODE={MANUAL|AUTOMATED}	Sets the global mode, as documented in the previous section.
WARM={YES|NO}	Specifies whether a warm load or a cold load is to be performed.

$RMCALL ACTION=DBGET

Syntax

```
$RMCALL     OPT=SERVICE
            SERVICE=ACTION
            ACTION=DBGET
            NAME=resource-name
            CLASS=cc
            SYSNAME=system-name
            VERSION=version
```

Use

Use this call to retrieve information about a resource from the knowledge base.

If you make this call from an NCL procedure, ensure that you share the &ZRMDB-prefixed variables, for example, by adding the following statement before the call:

```
&CONTROL SHRVARS=(ZRMDB)
```

Operands

Operand	Description
OPT=SERVICE	Indicates that an API service is to be performed.
SERVICE=ACTION	Indicates that an ACTION service is to be performed.
ACTION=DBGET	Indicates that database information about a resource is to be retrieved.
NAME=*resource-name*	Specifies the name of the resource for which information is to be retrieved.
CLASS=*cc*	Specifies the two-digit identifier of the ResourceView definition class (see page 383) to which the resource belongs.
SYSNAME=*system-name*	Specifies the name of the system image where the resource is defined.
VERSION=*version*	Specifies the version of the system image where the resource is defined.

Returned Variables

Variable	Description
&ZRMDB*	Reference definition information about the specified resource is returned in &ZRMDB* variables (see page 386).
&SYSMSG	Contains the message returned by $RMCALL.

Return Codes

The following return codes indicate the success or failure of command processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example retrieves information about the internal resource (class 21) named RES001, located on version 0001 of the EASTPRD1 system:

```
&CALL   PROC=$RMCALL +
        SHARE=(ZRMDB>) +
        PARMS=(OPT=SERVICE,+
               SERVICE=ACTION,+
               ACTION=DBGET,+
               NAME=RES001,+
               CLASS=21,+
               SYSNAME=EASTPRD1,+
               VERSION=0001)
```

More information:

$RMDBAPI SERVICE=GET (see page 349)

$RMCALL ACTION=PURGE

Syntax

```
$RMCALL     OPT=SERVICE
            SERVICE=ACTION
            ACTION=PURGE
            ACBNAME=acb-name
```

Use

When a linked region is decommissioned without first unlinking it, extraneous link records are left behind in other regions that were connected to it. You can use this call to delete these extraneous link records.

Operand

Operand	Description
ACBNAME=*acb-name*	The ACB name that identifies the link record.

Returned Variable

Variable	Description
&SYSMSG	Contains the message returned by $RMCALL.

Return Codes

The following return codes indicate the success or failure of command processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example deletes the SOLV1 link record:

```
&CALL    PROC=$RMCALL +
         PARMS=(OPT=SERVICE,+
                SERVICE=ACTION,+
                ACTION=PURGE,+
                ACBNAME=SOLV1)
```

$RMCALL ACTION=STGET

Syntax

```
$RMCALL          OPT=SERVICE
                 SERVICE=ACTION
                 ACTION=STGET
                 NAME=resource-name
                 CLASS=cc
                 SYSNAME=system-name
                 VERSION=version
```

Use

Use this call to retrieve information about the current status of a resource.

If you make this call from an NCL procedure, ensure that you share the &ZRMST-prefixed variables, for example, by adding the following statement before the call:

```
&CONTROL SHRVARS=(ZRMST)
```

Operands

Operand	Description
OPT=SERVICE	Indicates that an API service is to be performed.
SERVICE=ACTION	Indicates that an ACTION service is to be performed.
ACTION=STGET	Indicates that status information about a resource is to be retrieved.
NAME=*resource-name*	Specifies the name of the resource for which status information is to be retrieved.
CLASS=*cc*	Specifies the two-digit identifier of the ResourceView definition class (see page 383) to which the resource belongs.
SYSNAME=*system-name*	Specifies the name of the system image where the resource is defined.
VERSION=*version*	Specifies the version of the system image where the resource is defined.

Returned Variables

Variable	Description
&ZRMST*	Status information about the specified resource is returned in &ZRMST* variables (see page 390).
&SYSMSG	Contains the message returned by $RMCALL.

Return Codes

The following return codes indicate the success or failure of resource status processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example retrieves status information about the internal resource (class 21) named RES001, located on version 0001 of the EASTPRD1 system:

```
&CALL   PROC=$RMCALL +
        SHARE=(ZRMST>) +
        PARMS=(OPT=SERVICE,+
               SERVICE=ACTION,+
               ACTION=STGET,+
               NAME=RES001,+
               CLASS=21,+
               SYSNAME=EASTPRD1,+
               VERSION=0001)
```

$RMDBAPI SERVICE=CREATE

Syntax

```
$RMDBAPI        SERVICE=CREATE
                [TRUNCATE={YES|NO}]
                [{NAME=resource-name[MANNAME=manager-name]}|
                 {RSNAME=ft-ruleset-name[RMNAME=ft-rule-name]}]
                CLASS=cc
                SYSNAME=system-name
                VERSION=version
                [field-name-1=field value-1]
                [field-name-2=field value-2]
                .
                .
                .
                [field-name-n=field value-n]
```

Use

Use this call to create a ResourceView definition in the knowledge base.

Operands

Notes:

- Operand values must not contain the question mark (?) character.
- If possible, do not use the semi-colon (;) in values. By default, a semi-colon is interpreted as a command separator if the API is executed (EXEC) or started (START).
- If you must use the semi-colon and you are calling the API from an NCL procedure, use the &CALL verb.

Operand	Description
SERVICE=CREATE	Indicates that a definition is to be created in the knowledge base.
TRUNCATE={YES|NO}	Specifies whether a field value should be truncated if it is longer than the field length. ■ YES specifies that long values will be truncated and the definition will be created in the knowledge base. ■ NO specifies that no truncation is allowed. If a long value is encountered, the definition will not be created in the knowledge base.
NAME=*resource-name*	Specifies the name of the resource definition to be created (for example, the address of a DASD or the name of a started task). This operand is not required if you are creating a system image, a file transfer rule set, or a file transfer rule definition.
MANNAME=*manager-name* (FTM)	Specifies the name of the manager that owns the resource, *resource-name* (for example, the definition of the file transfer manager that owns the monitor resource whose definition is to be maintained).
RSNAME=*ft-ruleset-name* (FTM)	Specifies the name of the file transfer rule set to which the specified rule, *ft-rule-name*, belongs.
RMNAME=*ft-rule-name* (FTM)	Specifies the name of the file transfer rule definition to be created. This operand is not required if you are creating a file transfer rule set definition.

Operand	Description
CLASS=*cc*	Specifies the two-digit identifier of the ResourceView definition class (see page 383) to which the definition belongs.
SYSNAME=*system-name*	Specifies the name of the system image to be created or the name of the system image where the specified resource is to be created. **Note:** For file transfer rule sets and rules in a CA NetMaster FTM region, the value of SYSNAME is FILTER.
VERSION=*version*	Specifies the version of the system image. **Note:** For file transfer rule sets and rules in a CA NetMaster FTM region, the values of VERSION are 0006 and 0005, respectively.
field-name-n=field-value-n	Specifies the values of the fields in the definition.

More information:

System Image Fields (see page 339)
Resource Fields (see page 340)

Specify Field Values When Calling $RMDBAPI from an NCL Procedure

If you are making this call from an NCL procedure, consider the following:

- You can add the following statement before the call to share the field values:

  ```
  &CONTROL SHRVARS=(ZRMDB)
  ```

  Specify each value in ZRMDB*field-name-n*. In this case, you do not need to specify the field name operands. You can, however, override the specified variable values by using the operands.

  Note: If you share field values, the API ignores misspelled field names. If you misspell a name, the intended value will not be set.

- You can use the following statement to preserve case sensitive field values:

  ```
  &CONTROL NOUCASE
  ```

- If you use the operands to pass the values, beware of the following:
  - The maximum length of an &CALL or EXEC statement is 2048 characters.
  - You *must not* enclose values in quotes. Use variables to assign strings.
  - If a value contains blanks, then for an EXEC statement, the procedure must use &CONTROL NOVARSEG and the value must be passed as a variable, for example:

    ```
    &CONTROL NOVARSEG
    &DCMD=&STR D J,STC1
    EXEC $RMDBAPI SERVICE=CREATE … DISPCMD=&DCMD …
    ```

Specify Field Values When Submitting $RMDBAPI as a Command

If a value contains blanks, enclose the value in quotes.

Returned Variables

Variable	Description
&SYSMSG	Contains the message returned by $RMDBAPI.

Return Codes

The following return codes indicate the success or failure of the creation processing:

&RETCODE	Meaning
0	Processing was successful.
4	Processing was successful, but truncation has occurred.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example creates an EASTPRD1 version 2 system image definition (class 1), with truncation allowed:

```
&SDESC=&STR Eastern production system

&CALL   PROC=$RMDBAPI +
        PARMS=(SERVICE=CREATE,+
               CLASS=01,+
               SYSNAME=EASTPRD1,+
               VERSION=0002,+
               SDESC=&SDESC)
```

ResourceView Definition Field Names

The following sections list the field names for ResourceView definitions. For more information, see the product guides for any product specific field names not included here.

The names are related to the corresponding field labels on the appropriate definition panels:

- Fields that are mandatory on a panel are mandatory in the API.
- Values that are valid in the panel fields are valid in the API.
- Fields that have default values inherit the values in the API.

System Image Fields

This table lists the system image field names that can be used in the $RMDBAPI procedure.

Field Names	Field Label on Panel
System Image Definition	
HOMESYS	Home System
SDESC	Short Description
LDESC1 to LDESC4	Long Description
RULEID	EventView Ruleset to Activate

Resource Fields

The following table lists the resource field names that can be used in the $RMDBAPI procedure.

Field Names	Field Label on Panel
General Description	
TYPE	Type
OWNRNME	CICS Region Name (classes 4 to 7)
OWNRCLS	CICS Region Class (classes 4 to 7)
REMONME	Remote Region Name (class 7)
REMOCLS	Class (class 7)
REMOSMF	SMFID (class 7)
CRFILT	Containment Filter
LUNAME	LU Name (classes 11 and 14)
VOLUME	Volume (class 16)
ACBNAME	ACB Name
MODE	Operation Mode
SDESC	Short Description
LDESC1 to LDESC4	Long Description
TEMPLAT	TemplateName
TMPLACT	Template action code (M, O, or R)
Availability Map	
SCHED	Map Name

Field Names	Field Label on Panel
Group Filters (NA)	
RESNM1 to RESNM97	Resource Name
RESTP1 to RESTP97	Resource Type
FLTNM1 to FLTNM97	Filter Name
WGHT1 to WGHT97	Weight
WGHTT1 to WGHTT97	Weight Type
State Thresholds (NA)	
UNKTHR	UNKNOWN
FAILTHR	FAILED
DEGDTHR	DEGRADED
STOPTHR	STOPPING
INACTHR	INACTIVE
STRTTHR	STARTING
ACTTHR	ACTIVE
NOKTHR	Not OK
Activation Details	
INITCMD	System Command
INIMSGT	Expected Activation Completion Message
IN*x* (for example, INWILD)	See the field names for the panels from Define Extended Filter Definitions to Define Event Documentation in this table
INIPNAM	ProcessName
INIPRC1 to INIPRC2	Optional Parameters
INITIME	Timeout After
INITMST	On Timeout Assume Status of

Field Names	Field Label on Panel
Restart Control Parameters	
RETRYLM	Retry Attempt Limit
RETRYTM	Retry Time Limit
RETRYCM	System Command
RETRYPR	ProcessName
RETRYP1 to RETRYP2	Optional Parameters
Inactivation Details	
TRMTCMD	System Command
TRMMSGT	Expected Inactivation Completion Message
TR*x*	See the field names for the panels from Define Extended Filter Definitions to Define Event Documentation in this table
TRMPNAM	ProcessName
TRMPRC1 to TRMPRC2	Optional Parameters
TRMTIME	Timeout After
TRMTMST	On Timeout Assume Status of
TRMFRET	Try Force Inactivation
Force Inactivation Details	
FTRMCMD	System Command
FTRMSGT	Expected Force Inactivation Completion Message
FT*x*	See the field names for the panels from Define Extended Filter Definitions to Define Event Documentation in this table
FTRPNAM	ProcessName
FTRMPR1 to FTRMPR2	Optional Parameters
FTRTIME	Timeout After
FTRTMST	On Timeout Assume Status of

Field Names	Field Label on Panel
Display and Heartbeat Details	
DISPCMD	System Command
HBEATIN	Heartbeat Interval
ACTMSG	Message Text (ACTIVE status)
DA*x*	For ACTIVE status (see the field names for the panels Define Extended Filter Definitions to Define Event Documentation in this table)
INACMSG	Message Text (INACTIVE status)
DI*x*	For INACTIVE status (see the field names for the panels from Define Extended Filter Definitions to Define Event Documentation in this table)
STRTMSG	Message Text (STARTING status)
DS*x*	For STARTING status (see the field names for the panels from Define Extended Filter Definitions to Define Event Documentation in this table)
STOPMSG	Message Text (STOPPING status)
DP*x*	For STOPPING status (see the field names for the panels from Define Extended Filter Definitions to Define Event Documentation in this table)
DEGDMSG	Message Text (DEGRADED status)
DD*x*	For DEGRADED status (see the field names for the panels from Define Extended Filter Definitions to Define Event Documentation in this table)
FAILMSG	Message Text (FAILED status)
DF*x*	For FAILED status (see the field names for the panels from Define Extended Filter Definitions to Define Event Documentation in this table)
UNKMSG	Message Text (UNKNOWN status)

Field Names	Field Label on Panel
DU*x*	For UNKNOWN status (see the field names for the panels from Define Extended Filter Definitions to Define Event Documentation in this table)
DISPNAM	ProcessName
DISPRC1	Optional Parameters
Status Monitor Message Details	
MONMT01 to MONMT97	Messages
M01*x* to M97*x*	See the field names for the panels from Define Extended Filter Definitions to Define Event Documentation in this table
MONPR01 to MONPR97	Pty (specifies the priority of the message rule, which is used to determine which rule to use when a received message satisfies more than one rule)
MONST01 to MONST97	Status
State Change Exits	
SBAPROC	Process (before activation)
SBAPRM1 and SBAPRM2	Parameters (before activation)
STYPE1 to STYPE12	State Type
SFROM1 to SFROM12	Change From
STO1 to STO12	Change To
SPROC1 to SPROC12	Process
SPRM11 to SPRM112 SPRM21 to SPRM212	Parameters (first line) Parameters (second line)
Automation Log Details	
LOGTSIZ	Log Table Size
LOGAUTO	Log to Automation Log
LOGMVSC	Log to Console

Field Names	Field Label on Panel
LOGOCS	Log to OCS Window
LOGSYS	Log All System Msgs
LOGAUDT	Log Internal Audit Trail
Owner Details	
OWNNME1 and OWNNME2	Name
OWNUID1 and OWNUID2	Userid
OWNGRP1 and OWNGRP2	Group
OWNPHB1 and OWNPHB2	Phone Business Hours
OWNPHA1 and OWNPHA2	Phone After Hours
OWNPGR1 and OWNPGR2	Pager Number
Extended Function Exit	
FUNCNAM	Function Name
PARM1 to PARM12	Parameters
Define Extended Filter Definitions	
*x*WILD (for example, INWILD)	Wildcard Character
*x*SP1 to *x*SP5	Strt Pos
*x*WN1 to *x*WN5	Word Num
*x*OP1 to *x*OP5	Opr
*x*SC1 to *x*SC5	Scan Text
*x*EXP1 to *x*EXP5 (combined)	strt-pos,word-num,opr,scan-text
*x*RID	Expression
Define Event Related Actions	
*x*MOD	Mode

Field Names	Field Label on Panel
*x*ICMD	System Command
*x*RPLY	Reply (if WTOR)
*x*GALT	Generate Rsc Event
*x*LTXT	Log Message
*x*XPNM	ProcessName
*x*XPR1 and *x*XPR2	Optional Parameters
Define Event Exits	
*x*STXP	Process (State Change)
*x*STX1 and *x*STX2	Parameters (State Change)
*x*PRXP	Process (Problem)
*x*PRX1 and *x*PRX2	Parameters (Problem)
*x*GEXP	Process (General)
*x*GEX1 and *x*GEX2	Parameters (General)
Define Extended Display Attribute	
*x*EVDS	Extended Display (EXTDISP)
*x*INTN	Intensity
*x*COLR	Color
*x*HLIT	Highlight
*x*ICON	Use on Graphic Monitor?
*x*SEV	Severity
*x*FKWD	Keyword Value
*x*KWD1 to *x*KWD6	Var
*x*VAL1 to *x*VAL6	Value
Define Event Documentation	
*x*TMSG	Target Message
*x*NT1 to *x*NT12	Notes

$RMDBAPI SERVICE=DELETE

Syntax

```
$RMDBAPI      SERVICE=DELETE
              [{NAME=resource-name[MANNAME=manager-name]}|
               {RSNAME=ft-ruleset-name[RMNAME=ft-rule-name]}]
              CLASS=cc
              SYSNAME=system-name
              VERSION=version
```

Use

Use this call to delete a ResourceView definition from the knowledge base.

Operands

Operand	Description
SERVICE=DELETE	Indicates that a definition is to be deleted from the knowledge base.
NAME=*resource-name*	Specifies the name of the resource definition to be deleted. You cannot delete a resource that owns dependent resources (for example, a CICS started task that owns CICS resources). This operand is not required if you are deleting a system image, a file transfer rule set, or a file transfer rule.
MANNAME=*manager-name* (FTM)	Specifies the name of the manager that owns the resource, *resource-name* (for example, the definition of the file transfer manager that owns the monitor resource whose definition is to be deleted).
RSNAME=*ft-ruleset-name* (FTM)	Specifies the name of the file transfer rule set to be deleted or the name of the file transfer rule set from which the specified rule, *ft-rule-name*, is to be deleted. You cannot delete an active rule set. **Note:** When you delete a file transfer rule set, you also delete the rules it owns.

Operand	Description
RMNAME=*ft-rule-name* (FTM)	Specifies the name of the file transfer rule definition to be deleted. This operand is not required if you are deleting a file transfer rule set.
CLASS=*cc*	Specifies the two-digit identifier of the ResourceView definition class (see page 383) to which the definition belongs.
SYSNAME=*system-name*	Specifies the name of the system image to be deleted or the name of the system image from which the specified resource is to be deleted. **Notes:** ■ When you delete a system image, you also delete the resources it owns. ■ For file transfer rule sets and rules in a CA NetMaster FTM region, the value of SYSNAME is FILTER.
VERSION=*version*	Specifies the version of the system image. For file transfer rule sets and rules in a CA NetMaster FTM region, the values of VERSION are 0006 and 0005 respectively.

Returned Variable

Variable	Description
&SYSMSG	Contains the message returned by $RMDBAPI.

Return Codes

The following return codes indicate the success or failure of the deletion processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example deletes the RES001 printer definition (class 11) from the EASTPRD1 version 1 system image:

```
&CALL   PROC=$RMDBAPI +
        PARMS=(SERVICE=DELETE,+
               NAME=RES001,+
               CLASS=11,+
               SYSNAME=EASTPRD1,+
               VERSION=0001)
```

$RMDBAPI SERVICE=GET

Syntax

```
$RMDBAPI        SERVICE=GET
                [{NAME=resource-name[MANNAME=manager-name]}|
                 {RSNAME=ft-ruleset-name[RMNAME=ft-rule-name]}]
                CLASS=cc
                SYSNAME=system-name
                VERSION=version
```

Use

Use this call to retrieve information about a ResourceView definition in the knowledge base.

If you make this call from an NCL procedure, ensure that you share the &ZRMDB-prefixed variables, for example, by adding the following statement before the call:

```
&CONTROL SHRVARS=(ZRMDB)
```

Operands

Operand	Description
SERVICE=GET	Indicates that information about a definition is to be retrieved from the knowledge base.
NAME=*resource-name*	Specifies the name of the resource definition for which information is to be retrieved. This operand is not required if you are retrieving information about a system image definition, a file transfer rule set, or a file transfer rule.
MANNAME=*manager-name* (FTM)	Specifies the name of the manager that owns the resource, *resource-name* (for example, the definition of the file transfer manager that owns the monitor resource for which information is to be retrieved).
RSNAME=*ft-ruleset-name* (FTM)	Specifies the name of the file transfer rule set to which the specified rule, *ft-rule-name*, belongs.
RMNAME=*ft-rule-name* (FTM)	Specifies the name of the file transfer rule definition for which information is to be retrieved. This operand is not required if you are retrieving information about a file transfer rule set definition.
CLASS=*cc*	Specifies the two-digit identifier of the ResourceView definition class (see page 383) to which the definition belongs.
SYSNAME=*system-name*	Specifies the name of the system image for which information is to be retrieved or the name of the system image that owns the resource for which information is to be retrieved. For file transfer rule sets and rules in a CA NetMaster FTM region, the value of SYSNAME is FILTER.
VERSION=*version*	Specifies the version of the system image. For file transfer rule sets and rules in a CA NetMaster FTM region, the values of VERSION are 0006 and 0005 respectively.

Returned Variables

Variable	Description
&ZRMDB*field-name*	Knowledge base information about the specified resource is returned in &ZRMDB*field-name* variables.
&SYSMSG	Contains the message returned by $RMDBAPI.

Return Codes

The following return codes indicate the success or failure of the retrieval processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example retrieves information about the RES001 internal resource (class 21), located on the EASTPRD1 version 1 system image:

```
&CALL   PROC=$RMDBAPI +
        SHARE=(ZRMDB>) +
        PARMS=(SERVICE=GET,+
               NAME=RES001,+
               CLASS=21,+
               SYSNAME=EASTPRD1,+
               VERSION=0001)
```

More information:

ResourceView Definition Field Names (see page 339)

$RMDBAPI SERVICE=LIST

Syntax

```
$RMDBAPI        SERVICE=LIST
                [RSNAME=ft-ruleset-name]
                CLASS=cc
                [SYSNAME=system-name]
                [VERSION=version]
```

Use

Use this call to list ResourceView definitions in the knowledge base.

If you make this call from an NCL procedure, ensure that you share the &ZRMLST-prefixed variables, for example, by adding the following statement before the call:

```
&CONTROL SHRVARS=(ZRMLST)
```

Operands

Operand	Description
SERVICE=LIST	Indicates that selected definitions in the knowledge base are to be listed in *&ZRMLSTnnnn* variables.
RSNAME=*ft-ruleset-name* (FTM)	Specifies the name of the file transfer rule set that owns the rules to be listed. If you want to list the rule sets, do not specify this operand.
CLASS=*cc*	Specifies the two-digit identifier of the ResourceView definition class (see page 383) to which the definition belongs.
SYSNAME=*system-name*	Specifies the name of the system image that owns the resource definitions to be listed. **Note:** For file transfer rule sets and rules in a CA NetMaster FTM region, the value of SYSNAME is FILTER.
VERSION=*version*	Specifies the version of the system image. **Note:** For file transfer rule sets and rules in a CA NetMaster FTM region, the values of VERSION are 0006 and 0005 respectively.

Returned Variables

Variable	Description
&ZRMLST*nnnn*	Each knowledge base definition entry is returned in an &ZRMLST*nnnn* variable.
&SYSMSG	Contains the message returned by $RMDBAPI.

Return Codes

The following return codes indicate the success or failure of the list processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example retrieves the list of all versions of the EASTPRD1 system image:

```
&CALL   PROC=$RMDBAPI +
        SHARE=(ZRMLST>) +
        PARMS=(SERVICE=LIST,+
               CLASS=01,+
               SYSNAME=EASTPRD1)
```

$RMDBAPI SERVICE=SET

Syntax

```
$RMDBAPI        SERVICE=SET
                [TRUNCATE={YES|NO}]
                CLASS=01
                SYSNAME=system-name
                VERSION=version
                [field-name-1=field value-1]
                [field-name-2=field value-2]
                .
                .
                .
                [field-name-n=field value-n]
```

Use

Use this call to change field values in a system image definition. The SET function is not available to other definitions in the knowledge base.

Operands

Operand	Description
SERVICE=SET	Indicates that changes are to be made in the specified system image definition.
TRUNCATE={YES|NO}	Specifies whether a field value should be truncated if it is longer than the field length. ■ **YES** specifies that long values will be truncated and the definition will be changed. ■ **NO** specifies that no truncation is allowed. If a long value is encountered, the definition will not be changed.
CLASS=01	Indicates that a system image definition is to be changed.
SYSNAME=*system-name*	Specifies the name of the system image definition to be changed.
VERSION=*version*	Specifies the version of the system image definition.

Operand	Description
field-name-n=field-value-n	Specifies the values of the fields to be changed in the definition. You cannot delete a field value.

Returned Variable

Variable	Description
&SYSMSG	Contains the message returned by $RMDBAPI.

Return Codes

The following return codes indicate the success or failure of the change processing:

&RETCODE	Meaning
0	Processing was successful.
4	Processing was successful, but truncation has occurred.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example changes the value of the fourth line of the Long Description field in the EASTPRD1 version 2 system image definition (class 1), with truncation allowed:

```
&LDESC4=&STR (Upgrade in progress)

&CALL   PROC=$RMDBAPI +
        PARMS=(SERVICE=SET,+
               CLASS=01,+
               SYSNAME=EASTPRD1,+
               VERSION=0002,+
               LDESC4=&LDESC4)
```

$RMEVENT

Syntax

```
$RMEVENT        CLASS={ USRCLS | class-name }
                NAME=resource-name
                MSG='message-text '
```

Use

Use the $RMEVENT API procedure to send a message to a defined resource. The resource definition must interpret the message and control the actual state of the resource. If necessary, relevant actions are invoked as part of this process. This procedure is typically used to control user-defined resources.

Operands

Operand	Description
CLASS={ USRCLS | *class-name* }	The class of the resource that is the subject of the message. The default, USRCLS, specifies a user-defined resource class.
NAME=*resource-name*	The name of the resource that is the subject of the message.
MSG='*message-text*'	The text of the message that is sent to the resource. The message text must be enclosed in quotation marks.

Returned Variable

Variable	Description
&SYSMSG	Contains the message returned by $RMEVENT.

Return Codes

The following return codes indicate the success or failure of processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example sends a user-defined message (that is, a message defined in the USRCLS class) to the resource named MYRES01:

```
$RMEVENT NAME=MYRES01 MSG='RECOVER'
```

$RMSTSET

Syntax

```
$RMSTSET        CLASS={USRCLS |class-name}
                NAME=resource-name
                STATUS=actual-state
```

Use

Use this API to set the actual state of a defined resource.

Operands

Operand	Description
CLASS={ USRCLS | *class-name* }	The class of the resource that is the subject of the status change. USRCLS specifies the user-defined resource class.
NAME=*resource-name*	The resource that is the subject of the status change.
STATUS=*actual-state*	The actual state to which the resource is to be set.

Returned Variable

Variable	Description
&SYSMSG	Contains the message returned by $RMSTSET.

Return Codes

The following return codes indicate the success or failure of command processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example sets the actual state of MYRES01 to FAILED:

```
$RMSTSET NAME=MYRES01 STATUS=FAILED
```

$RECALL SERVICE=SET

Syntax

```
$RECALL         SERVICE=SET
                CLASS=VARIABLE
                NAME='VARNAME=variable-name'
                PARMS='VALUE=new-value'
                [DESC=value-description]
```

Use

Use this API to set the value of an EventView variable.

Operands

Operand	Description
SERVICE=SET	A value is to be set.
CLASS=VARIABLE	The value to be set is an EventView variable.
NAME='VARNAME=*variable-name*'	Identifies the EventView variable. *variable-name* can be up to eight characters long.
PARMS='VALUE=*new-value*'	The new value to be set for the specified variable.
DESC=*value-description*	Optional entry to describe the variable value.

Returned Variable

Variable	Description
&SYSMSG	Contains the message returned by $RECALL.

Return Codes

The following return codes indicate the success or failure of command processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example sets the value of TEST:

```
&VALUE = &ZQUOTE Test's value at &TIME
&DESC = &ZQUOTE Why Test was changed
&PARMS = &ZQUOTE VALUE=&VALUE DESC=&DESC
&CALL PROC=$RECALL +
      PARMS=(SERVICE=SET,+
             CLASS=VARIABLE,+
             NAME='VARNAME=TEST',+
             PARMS=&PARMS)
```

$RECALL SERVICE=GET

Syntax

```
$RECALL         SERVICE=GET
                CLASS=VARIABLE
                NAME='VARNAME=variable-name'
```

Use

Use this API to retrieve the value of an EventView variable.

If you make this call from an NCL procedure, ensure that you share the &$REVAR-prefixed variables, for example, by adding the following statement before the call:

```
&CONTROL SHRVARS=($REVAR)
```

Operands

Operand	Description
SERVICE=GET	A value is to be retrieved.
CLASS=VARIABLE	The value to get is the value of an EventView variable.
NAME='VARNAME=*variable-name*'	Identifies the EventView variable.

Returned Variables

Variable	Description
&$REVARNAME	Contains the name of the variable.
&$REVARVALUE	Contains the value of the variable.
&$REVARDESC	Contains a description of the value of the variable.
&$REVARSTATS	Contains the date, time, and the user ID of the user who last updated the value.
&SYSMSG	Contains the message returned by $RECALL.

Return Codes

The following return codes indicate the success or failure of command processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example retrieves the value of TEST:

```
&CALL PROC=$RECALL SHARE=($RE>) +
      PARMS=(SERVICE=GET,+
             CLASS=VARIABLE,+
             NAME='VARNAME=TEST')
&WRITE ZREVTEMP=&$REVARVALUE
```

$RECALL SERVICE=ACTION

Syntax

```
$RECALL         SERVICE=ACTION
                ACTION={ACT | INACT}
                NAME='RULESET=rulesetname'
```

Use

Use this API to activate or inactivate a rule set.

Operands

Operand	Description
SERVICE=ACTION	An action is to be performed on a rule set.
ACTION={ ACT | INACT }	The rule set is to be activated or inactivated.
NAME='RULESET=*rulesetname'*	Names the rule set to be activated or inactivated.

Returned Variable

Variable	Description
&SYSMSG	Contains the message returned by $RECALL.

Return Codes

The following return codes indicate the success or failure of command processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Examples

The following example activates rule set SET01:

```
$RECALL SERVICE=ACTION ACTION=ACT NAME='RULESET=SET01'
```

The following example inactivates rule set SET01:

```
$RECALL SERVICE=ACTION ACTION=INACT NAME='RULESET=SET01'
```

$REDBAPI SERVICE=CREATE

Syntax

```
$REDBAPI        SERVICE=CREATE
                [TRUNCATE={YES|NO}]
                CLASS=cc
                RULESET=ruleset-name
                [field-name-1=field value-1]
                [field-name-2=field value-2]
                .
                .
                .
                [field-name-n=field value-n]
```

Use

Use this call to create an EventView rule set or message rule definition in the knowledge base.

Operands

Important! If possible, do not use the semi-colon (;) in values. By default, a semi-colon is interpreted as a command separator if the API is executed (EXEC) or started (START). If you must use the semi-colon and you are calling the API from an NCL procedure, use the &CALL verb.

Operand	Description
SERVICE=CREATE	Indicates that a definition is to be created in the knowledge base.
TRUNCATE={YES|NO}	Specifies whether a field value should be truncated if it is longer than the field length. ■ **YES** specifies that long values will be truncated and the definition will be created in the knowledge base. ■ **NO** specifies that no truncation is allowed. If a long value is encountered, the definition will not be created in the knowledge base.
CLASS=*cc*	Specifies the two digit identifier of the EventView class to which the definition belongs as follows: 93 for rule sets and 94 for message rules.
RULESET=*ruleset-name*	Specifies the name of the rule set to be created or the name of the rule set where a message rule is to be created.
field-name-n=field-value-n	Specifies the values of the fields in the definition. A value must not contain the question mark (?) character.

More information:

Rule Set Fields (see page 367)
Message Rule Fields (see page 368)

Specify Field Values When Calling $REDBAPI from an NCL Procedure

If you are making this call from an NCL procedure, consider the following:

- You can add the following statement before the call to share the field values:

  ```
  &CONTROL SHRVARS=(ZRE)
  ```

  Specify each value in ZRE*field-name-n*. In this case, you do not need to specify the field name operands. You can, however, override the specified variable values by using the operands.

  Important! If you share field values, the API ignores misspelled field names. If you misspell a name, the intended value is not set.

- You can use the following statement to preserve case-sensitive field values:

  ```
  &CONTROL NOUCASE
  ```

- If you use the operands to pass the values, beware of the following:
  - The maximum length of an &CALL or EXEC statement is 2048 characters.
  - You *must not* enclose values in quotes. Use variables to assign strings.
  - If a value contains blanks, then for an EXEC statement, the procedure must use &CONTROL NOVARSEG and the value must be passed as a variable, for example:

    ```
    &CONTROL NOVARSEG
    &DESC=&STR SUPPRESSION RULES
    EXEC $REDBAPI SERVICE=CREATE … SDESC=&DESC …
    ```

Specify Field Values When Submitting $REDBAPI as a Command

If a value contains blanks, enclose the value in quotes.

Returned Variables

Variable	Description
&ZRENAME	Contains the name of a created message rule.
&SYSMSG	Contains the message returned by $REDBAPI.

Return Codes

The following return codes indicate the success or failure of the creation processing:

&RETCODE	Meaning
0	Processing was successful.
4	Processing was successful, but truncation has occurred.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example creates the SUPP rule set definition (class 93), with truncation allowed:

```
&SDESC=&STR Suppression rules

&CALL    PROC=$REDBAPI +
         PARMS=(SERVICE=CREATE,+
                CLASS=93,+
                RULESET=SUPP,+
                SDESC=&SDESC)
```

EventView Definition Field Names

The following sections list the field names for EventView rule sets and message rules.

The names are related to the corresponding field labels on the appropriate definition panels:

- Fields that are mandatory on a panel are mandatory in the API
- Values that are valid in the panel fields are valid in the API.
- Fields that have default values inherit the values in the API.

Rule Set Fields

This table lists the rule set field names that can be used in the $REDBAPI procedure.

Field Names	Field Label on Panel
Rule Set Description	
RULSTAT	Ruleset Status
SDESC	Short Description
RSDELIV	Default Message Delivery
RSMOD	Perform Message Modification?
RSACT	Perform Action?
RSLOG	Log Ruleset Activity?
RSSTAT	Collect Statistics?
RSLEARN	Learn New Messages?
Rule Set Comments	
COMMENT1 to COMMENT12	Comment Text

Message Rule Fields

This table lists the message rule field names that can be used in the $REDBAPI procedure.

Field Names	Field Label on Panel
Message Filter	
RULSTAT	Rule Status
SDESC	Short Description
TSTTXT	Message Text
JOBNAME	Job Name
EJOBTYPE	Job Type
RULEPRI	Rule Priority
BESTFIT	Execute if not Best Fit?
DAYMAP (*xxxxxxx*) or DAY1 to DAY7	On Days
TSTART1 to TSTART2	Range*n* Start
TEND1 to TEND2	Range*n* End
Extended Message Filter	
EWILDC	Wildcard Character
EDESCCD	Descriptor Code
EROUTCD	Route Code
EMSGID	Message ID
MVSSYS	System Name
ESTPOS1 to ESTPOS5	Strt Pos
EWORD1 to EWORD5	Word Num
EOPER1 to EOPER5	Opr
ETXT1 to ETXT5	Scan Text
EEXPR	Expression

Field Names	Field Label on Panel
Set Test Variables	
TSTVAR1 to TSTVAR6 (*name*=*value*) or TVAR1 and TVALUE1 to TVAR6 and TVALUE6	Name and Value
Message Delivery	
MDELIV	Deliver
THRSMAX	Maximum Number
THRSINT	Time Interval
THRSACT	Do Action
THRSCOR	Correlation Key
Message Modification	
MSGTXT	Replacement Text
MDESCCD	Set Descriptor Code
MROUTCD	Set Route Code
COLOR	Color
HLITE	Highlight
INTENS	Intensity
MON	Monitor?
ALARM	Alarm?
NRD	NRD?
MSGCODE	Message Code
Message Actions	
REPLTXT	Reply Text
SYSCMD	System Command
SOLVCMD	MS Command
SSOPROC (combined) or SSOPNAME and SSOPPRM	Process and Parameters
SSOCMD (combined) or SSOCNAME and SSOCPRM	Command and Parameters

Field Names	Field Label on Panel
Related Message Groups	
MGRPID1 to MGRPID5	Group Name
MCOR1 to MCOR5	Correlation Key
Set Variables	
SETVAR1 to SETVAR6 (*name=value*) or VAR1 and VALUE1 to VAR6 and VALUE6	Name and Value
Rule Comments	
COMMENT1 to COMMENT12	Comment Text

$REDBAPI SERVICE=DELETE

Syntax

```
$REDBAPI        SERVICE=DELETE
                [NAME=rule-object-name]
                CLASS=cc
                RULESET=ruleset-name
```

Use

Use this call to delete an EventView rule set or message rule definition from the knowledge base.

Operands

Operand	Description
SERVICE=DELETE	Indicates that a definition is to be deleted from the knowledge base.
NAME=*rule-object-name*	Specifies the object name of the message rule definition to be deleted. The name is contained in the &ZRENAME variable returned by a previous $REDBAPI SERVICE=CREATE call. This operand is not required if you are deleting a rule set.
CLASS=*cc*	Specifies the two digit identifier of the EventView class to which the definition belongs as follows: 93 for rule sets and 94 for message rules.
RULESET=*ruleset-name*	Specifies the name of the rule set to be deleted or the name of the rule set that owns the message rule is to be deleted. When you delete a rule set, you also delete the rules it owns.

Returned Variable

Variable	Description
&SYSMSG	Contains the message returned by $REDBAPI.

Return Codes

The following return codes indicate the success or failure of the deletion processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example deletes the SUPP rule set (class 93):

```
&CALL   PROC=$REDBAPI +
        PARMS=(SERVICE=DELETE,+
               CLASS=93,+
               RULESET=SUPP)
```

$REDBAPI SERVICE=GET

Syntax

```
$REDBAPI        SERVICE=GET
                [NAME=rule-object-name]
                CLASS=cc
                RULESET=ruleset-name
```

Use

Use this call to retrieve information about an EventView rule set or message rule definition in the knowledge base.

If you make this call from an NCL procedure, ensure that you share the &ZRE-prefixed variables, for example, by adding the following statement before the call:

```
&CONTROL SHRVARS=(ZRE)
```

Operands

Operand	Description
SERVICE=GET	Indicates that information about a definition is to be retrieved from the knowledge base.
NAME=*rule-object-name*	Specifies the object name of the message rule definition for which information is to be retrieved. The name is contained in the &ZRENAME variable returned by a previous $REDBAPI SERVICE=CREATE call. This operand is not required if you are retrieving information about a rule set definition.
CLASS=*cc*	Specifies the two digit identifier of the EventView class to which the definition belongs as follows: 93 for rule sets and 94 for message rules.

Operand	Description
RULESET=*ruleset-name*	Specifies the name of the rule set for which information is to be retrieved or the name of the rule set that owns the message rule for which information is to be retrieved.

Returned Variables

Variable	Description
&ZRE*field-name*	Knowledge base information about the specified rule set or rule is returned in &ZRE*field-name* variables.
&SYSMSG	Contains the message returned by $REDBAPI.

Return Codes

The following return codes indicate the success or failure of the retrieval processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example retrieves information about a message rule (class 94) previously created in the SUPP rule set:

```
&CALL PROC=$REDBAPI PARMS=(SERVICE=CREATE,…)
.
.
.
&JAA992I=&ZRENAME
.
.
.
&CALL   PROC=$REDBAPI +
        SHARE=(ZRE>) +
        PARMS=(SERVICE=GET,+
               NAME=&JAA992I,+
               CLASS=94,+
               RULESET=SUPP)
```

More information:

EventView Definition Field Names (see page 367)

$REDBAPI SERVICE=LIST

Syntax

```
$REDBAPI       SERVICE=LIST
               CLASS=cc
               [RULESET=ruleset-name]
               [RULEID=message-rule-id]
```

Use

Use this call to list EventView rule set or message rule definitions in the knowledge base.

If you make this call from an NCL procedure, ensure that you add the following statement before the call:

```
&CONTROL SHRVARS=(ZRELST)
```

Operands

Operand	Description
SERVICE=LIST	Indicates that selected definitions in the knowledge base are to be listed in &ZRELST*nnnn* variables.
CLASS=*cc*	Specifies the two digit identifier of the EventView class to which the definitions belong as follows: 93 for rule sets and 94 for message rules.
RULESET=*ruleset-name*	Identifies the rule sets to be listed, or specifies the name of the rule set that owns the message rules to be listed. If you are listing rule sets, the value is generic. For example, if RULESET=SUP, then rule sets with names that start with SUP are matched.
RULEID=*message-rule-id*	Specifies the IDs of the message rules to list. The ID is the first word of the message text. The value is generic. For example, if RULEID=JBB1, then message rules with a first word that starts with JBB1 are matched.

Returned Variables

Variable	Description
&ZRELST*nnnn*	Each knowledge base definition entry is returned in an &ZRELST*nnnn* variable.
&SYSMSG	Contains the message returned by $REDBAPI.

Return Codes

The following return codes indicate the success or failure of the list processing:

&RETCODE	Meaning
0	Processing was successful.
8	Processing failed.
16	Error occurred in call syntax.

Example

The following example retrieves the list of all rule sets:

```
&CALL    PROC=$REDBAPI +
         SHARE=(ZRELST>) +
         PARMS=(SERVICE=LIST,+
                CLASS=93)
```

Appendix G: Automation Services SMF Record Format

This section contains the following topics:

Automation Services SMF Records (see page 377)
SMF Header Format (see page 378)
How to Obtain User SMF Record Types (see page 378)
EventView SMF Record Format (see page 379)
ResourceView and ServiceView Record Format (see page 380)
User-defined Record Format (see page 381)

Automation Services SMF Records

The following types of SMF records are written by Automation Services:

EventView SMF record

Contains two formats; the first is used to indicate the start of statistics collection, the second contains the actual data collected by EventView.

ResourceView and ServiceView SMF record

Contains data collected about resources or services. One record is written for each resource or service.

User-defined record

Contains a number of user-specified formats that contain data written by the SMFWRITE macro.

SMF Header Format

SMF records described in this appendix are written by Automation Services and have a header of the format shown in the following table:

Offset		Position	Contains...
Dec	**Hex**		
0	0	1-18	SMF record header
18	12	19-22	Subsystem identifier
22	16	23-24	The SMF record subtype
24	18	25-36	The region ID
36	24	37-*n*	The data to be written

How to Obtain User SMF Record Types

You should obtain user SMF record types (128-255) for both old (unsubtyped) and new (subtyped) records. To do this, contact the System Programming group and ask them to do the following:

1. Assign SMF record types that do not conflict with other users or applications.
2. Make sure that those numbers are not excluded by SMF.
3. Make sure that installed SMF exits do not ignore assigned SMF record types, that is, they are not written to SMF data sets.

EventView SMF Record Format

The EventView SMF record that indicates the start of statistics collection has a subtype of 2000 (hexadecimal value). This subtype has no data associated with it.

The EventView SMF record that contains the EventView statistics data has a subtype of 2200 (hexadecimal value). All fields in the record are four bytes long.

The following table lists the fields in this record:

Offset		Name	Length	Format	Description
Dec	Hex				
36	24	MSGCOUNT	4	BINARY	Number of messages processed
40	28	SUPPCOUNT	4	BINARY	Number of messages suppressed
44	2C	MODCOUNT	4	BINARY	Number of messages modified
48	30	REPCOUNT	4	BINARY	Number of replies issued to WTORs
52	34	SYSTEMCMD	4	BINARY	Number of system commands issued
56	38	SOLVECMD	4	BINARY	Number of system services commands issued
60	3C	SSOPRCSS	4	BINARY	Number of Automation Services processes issued
64	40	SSOCMD	4	BINARY	Number of Automation Services commands issued
68	44	VARSET	4	BINARY	Number of variables set
72	48	TMRPOP	4	BINARY	Number of timers that were triggered
76	4C	ISSUEMSG	4	BINARY	Number of messages issued

ResourceView and ServiceView Record Format

The ResourceView and ServiceView SMF record has a subtype of 3000 (hexadecimal value).

The following table shows the fields in this record:

Offset		Name	Length	Format	Description
Dec	Hex				
36	24	SYSNAME	8	EBCDIC	System Image Name
44	2C	SYSVERSION	4	EBCDIC	System Image Version
48	30	RESCLASS	2	EBCDIC	Resource Class
50	32	NAME	18	EBCDIC	Resource or Service Name
68	44	AVIDATE	12	EBCDIC	Resource initialization date
80	50	AVITIME	12	EBCDIC	Resource initialization time
92	5C	AVDATE	4	BINARY	Date
96	60	AVTIME	12	EBCDIC	Time
108	6C	AVSTATUS	12	EBCDIC	Current status
120	78	AVATIME	4	BINARY	Total time available (in minutes)
124	7C	AVUTIME	4	BINARY	Total time unavailable (in minutes)
128	80	AVACOUNT	4	BINARY	Number of times the resource became available
132	84	AVUCOUNT	4	BINARY	Number of times the resource became unavailable
136	88	AVPMSGS	4	BINARY	Number of messages processed for the resource
140	8C	AVPCMDS	4	BINARY	Number of commands issued for the resource

User-defined Record Format

A user-defined record SMF format has a subtype of 9*xxx* (hexadecimal value), where *xxx* is specified in the SMFWRITE macro.

The user defines the data fields in the record.

Appendix H: RAMDB Definition Classes

This section contains the following topics:

Definition Classes (see page 383)

Definition Classes

ResourceView definitions created in the knowledge base are differentiated by classes.

The following table lists the common definition classes. Classes that are specific to a particular product are identified in parentheses, where:

- FT specifies CA NetMaster FTM.
- IP specifies CA NetMaster NM for TCP/IP.
- SNA specifies CA NetMaster NM for SNA.
- NA specifies CA NetMaster NA.

Definition	Class Name	Class Number
APPN/HPR (IP)	APPNHPR	53
CA-XCOM manager (FT)	XCMGR	62
CA-XCOM monitor (FT)	XCMON	63
Channel Interface Processor (IP)	CIP	44
Communications storage manager (IP)	CSM	48
CONNECT:Direct manager (FT)	CDMGR	08
CONNECT:Direct monitor (FT)	CDMON	09
CONNECT:Mailbox manager (FT)	CMMGR	27
CONNECT:Mailbox monitor (FT)	CMMON	28
Enterprise Extender (IP)	EE	47
File transfer schedule monitor (FT)	FTSCHD	24
FTP manager (FT)	FTPMGR	25
FTP monitor (FT)	FTPMON	26

Definition	Class Name	Class Number
FTS manager (FT)	FTSMGR	22
FTS monitor (FT)	FTSMON	23
Internal resource	INTNL	21
IP application (IP)	ASMON	43
Network control point monitor (SNA)	NCPMON	54
Open Systems Adapter (IP)	OSA	46
Service	SVC	61
SNA group (NA)	SNAGRP	03
TCP/IP stack (IP)	STACK	51
User class resource	USRCLS	17

Appendix I: RAMDB Variables

This section contains the following topics:

Variable Types (see page 385)

Variable Types

Variables let you access knowledge base data, find out about the status of services and resources, and extract information about messages.

Variables comprise the following types:

- Variables that contain data in the knowledge base
- Variables that contain status information
- Variables that contain message information

This appendix describes these variables. The variables that are available depend on the product you are using.

Note: For information about other variables that are specific to your product, see the *Administration Guide*.

Knowledge Base Variables

Use the following variables to retrieve data from the knowledge base. Variables that are specific to a particular product are identified in parentheses.

You can use the variables to pass values to the fields in a template, to the fields in a service or resource definition, to the parameters in macros used in the definition, and to NCL procedures invoked from the definition.

&ZRMDBCLASS

Contains the class number.

&ZRMDBCLDESCS

Contains the class name.

&ZRMDBREL1 to &ZRMDBREL25

Contain the names of the immediate parents.

&ZRMDBSCHED

Contains the name of the availability map.

&ZRMDBSYSNAME

Contains the name of the system image.

&ZRMDBVERSION

Contains the version of the system image.

Description Panel Fields

The following fields, if applicable, are displayed on the Description panel of a service or resource definition:

&ZRMDBACBNAME (CICS and z/OS)

Contains the value of the ACB Name field.

&ZRMDBACTNAME

Contains the same value as &ZRMDBNAME; however, if the value is qualified, the qualifier is dropped. For example, a CDMON resource name has the owner as the qualifier, owner-name.resource-name. This variable will return only resource-name.

&ZRMDBMODE

Contains the value of the Operation Mode field. The value can be AUTOMATED, IGNORED, MANUAL, or OFF.

&ZRMDBNAME

Contains the value of the field that identifies the service or resource.

&ZRMDBOWNRCL#

Contains the class number of the resource owner.

&ZRMDBOWNRCLS

Contains the value of the field that identifies the class name of the resource owner.

&ZRMDBOWNRNME

Contains the value of the field that identifies the owner of the resource.

&ZRMDBSDESC

Contains the value of the Short Description field.

&ZRMDBTEMPLAT

Contains the value of the TemplateName field.

&ZRMDBTYPE

Contains the value of the Type field.

Activation and Inactivation Details Fields

The following field is displayed on the Activation Details panel of a resource definition:

&ZRMDBINITCMD

Contains the value of the Command field.

The following field is displayed on the Inactivation Details panel of a resource definition:

&ZRMDBTMRTCMD

Contains the value of the Command field.

The following field is displayed on the Force Inactivation Details panel of a resource definition:

&ZRMDBFTRMCMD

Contains the value of the Command field.

Display and Heartbeat Field

The following field is displayed on the Display and Heartbeat Details panel of a resource definition:

&ZRMDBDISPCMD

Contains the value of the Command field.

Automation Log Details Fields

The following fields are displayed on the Automation Log Details panel of a service or resource definition:

&ZRMDBLOGAUDT

Contains the value of the Log Internal Audit Trail field.

&ZRMDBLOGMVSC

Contains the value of the Log to Console field.

&ZRMDBLOGOCS

Contains the value of the Log to OCS Window field.

&ZRMDBLOGRAM

Contains the value of the Log to Automation Log field.

&ZRMDBLOGSYS

Contains the value of the Log All System Msgs field.

&ZRMDBLOGTSIZ

Contains the value of the Log Table Size field.

First Level Support Details Fields

The following fields are displayed under First Level Support Details on the Owner Details panel of a service or resource definition:

&ZRMDBREOGRP1

Contains the value of the Group field.

&ZRMDBREOPAG1

Contains the value of the Pager Number field.

&ZRMDBREOPHA1

Contains the value of the Phone After Hours field.

&ZRMDBREOPHB1

Contains the value of the Phone Business Hours field.

&ZRMDBREOWNN1

Contains the value of the Name field.

&ZRMDBRESOID1

Contains the value of the Userid field.

Second Level Support Details Fields

The following fields are displayed under Second Level Support Details on the Owner Details panel of a service or resource definition:

&ZRMDBREOGRP2

Contains the value of the Group field.

&ZRMDBREOPAG2

Contains the value of the Pager Number field.

&ZRMDBREOPHA2

Contains the value of the Phone After Hours field.

&ZRMDBREOPHB2

Contains the value of the Phone Business Hours field.

&ZRMDBREOWNN2

Contains the value of the Name field.

&ZRMDBRESOID2

Contains the value of the Userid field.

Status Variables

Use the variables in the following table to retrieve the status of a service or resource.

You can use the variables in the same places that you use knowledge base variables.

&ZRMSTACTSTAT

Contains the base actual state only. The &ZRMSTPHYSTAT can contain any of the detectable actual states. The value can be ACTIVE, INACTIVE, STARTING, STOPPING, or UNKNOWN.

&ZRMSTACTDESC

Contains information about the actual state in the &ZRMSTPHYSTAT variable (for example, the message that caused the state to be set). The information is displayed on the Modes and States panel when an S (Status) command is executed.

&ZRMSTACTSTA0

Contains information about the base actual state in the &ZRMSTACTSTAT variable (for example, the message that caused the state to be set).

&ZRMSTARMFLAG

Contains information that indicates whether a resource is registered to the automatic restart manager (ARM) or if ARM is restarting the resource. The value can be YES, NO, or RECOVERY.

&ZRMSTAUTSTAT

Contains the automation status. The value can be FAILED, IN PROGRESS, or NONE. The information is displayed on the Modes and States panel when an S (Status) command is executed.

&ZRMSTCLASS

Contains the class number (for example, 17 for the user class).

&ZRMSTCLDESCL

Contains the class description.

&ZRMSTCLSNAME

Contains the class name.

&ZRMSTSDESC

Contains the short description of the resource as specified in the Short Description field of the resource definition.

&ZRMSTDESSTAT

Contains the desired state scheduled by the availability map. The value can be ACTIVE, INACTIVE, or RESET.

&ZRMSTDESSTA0

Contains information about the scheduled desired state (for example, the time when the state is set).

&ZRMSTDSTACUR

Contains the current desired state. The value can be ACTIVE or INACTIVE.

&ZRMSTDSTACU0

Contains information about the current desired state (for example, whether the state is scheduled or overridden).

&ZRMSTDSTAOV

Contains the desired state override. The value can be ACTIVE, INACTIVE, or NONE.

&ZRMSTDYNAMIC

Contains the value that indicates whether the definition for the resource is dynamic. The value can be NO or YES.

&ZRMSTEXIT

Contains the value that indicates whether a state change exit process exists. The value can be NO or YES.

&ZRMSTEXTDISP

Contains any activated extended display.

&ZRMSTMODECUR

Contains the current operation mode. The value can be AUTOMATED, IGNORED, or MANUAL.

&ZRMSTMODECU0

Contains information about the current operation mode (for example, whether the mode is overridden).

&ZRMSTMODEDB

Contains the operation mode specified in the definition. The value can be AUTOMATED, IGNORED, OFF, or MANUAL.

&ZRMSTMODEMAP

Contains the operation mode scheduled by the availability map. The value can be AUTOMATED, IGNORED, MANUAL, NONE, or RESET.

&ZRMSTMODEOV

Contains the operation mode override. The value can be AUTOMATED, IGNORED, MANUAL, or NONE.

&ZRMSTNAME

Contains the name of the resource or service.

&ZRMSTNMDID

Contains the domain ID of the region that monitors and controls the resource.

&ZRMSTNRMSTAT

Contains the logical state. The value can be ATTENTION, DEGRADED, FAILED, INERROR, OK, PENDING, STARTING, STOPPING, or UNKNOWN.

&ZRMSTOBJID

Contains the object ID of the record. The object ID is the concatenation of system image name, system image version, class number, and name.

&ZRMSTPHYSTAT

Contains the actual state. The value can be ACTIVE, DEGRADED, FAILED, INACTIVE, STARTING, STOPPING, or UNKNOWN.

&ZRMSTSYSNAME

Contains the name of the system image to which the resource belongs.

&ZRMSTTYPE

Contains the resource type as specified in the Type field of the resource definition.

Active System Image Variables

The following variables return the name and version of the local active system image:

- &ZRMSYSNAME
- &ZRMVERSION

Message Variables

Use these variables to retrieve information about a message. Some of the variables have equivalent AOM NCL system variables that are described in more detail in the *Network Control Language Reference Guide*.

You can use the variables to pass values to a message rule, to the parameters of macros used in the rule, and to NCL procedures invoked from the rule.

Note: The special messages prefixes ($AA-, $DN-, or $MN-) may be used in ResourceView to enhance functionality. A message captured by using such a prefix has the prefix included in the message variables.

&ZMSGATEXT (&AOMATEXT)

Contains the text of the current line of a message.

&ZMSGAWORD*n*

Contains a word in the current line of a message. *n* is a number indicating the position of the word in the message and is in the range 1 through 32. Words are delimited by a blank or a comma.

&ZMSGJOBID (&AOMJOBID)

Contains the identification number of the job that sends the message.

&ZMSGJOBNM (&AOMJOBNM)

Contains the name of the job that issued the message. The value may be overridden by a job name supplied by JES. Thus, messages originating from JES can have the name of the target job, rather than the JES job name.

&ZMSGMSGID (&AOMMSGID)

Contains the first word of the message text and is equivalent to &ZMSGWORD1. If the message is a multiline WTO or WTOR message, &ZMSGMSGID contains the first word of the major line text.

&ZMSGTEXT (&AOMTEXT)

Contains the text of the message. If the message is a multiline WTO or WTOR message, &ZMSGTEXT contains the first (major) line of the message. Use the &ZMSGATEXT variable to retrieve the current line (which can be a minor line) of a message.

&ZMSGTIME

Contains the time of message arrival, in the form *hh*.*mm*.*ss*. The value can be in the range 00.00.00 through 23.59.59.

&ZMSGWORD*n*

Contains a word in the message text. If the message is a multi-line WTO or WTOR message, it is treated as one long string and ZMSGWORD*n* variables are assigned accordingly. Words are delimited by a blank or a comma. Use the &ZMSGAWORD*n* variable to retrieve a word in the current line (which can be a minor line) of a message.

&ZMSGWRID (&AOMWRID)

Contains the WTOR reply number. The value is blank if the message is not of the WTOR type.

Variables Available to EventView Message Group Rules

The following variables are available to EventView message rules only:

&ZMSGIJOBN (&AOMIJOBN)

Contains the name of the job that issued the WTO or WTOR message.

&ZMSGSYSNM

Contains the name of the system from which the message originates.

Variables Available to Completed EventView Message Group Rules

The following variables are available to completed (or triggered) EventView message group rules:

&ZMSGCORR

Contains the correlation key used for the triggered message group rule.

&ZMSGGRP

Contains the name of the triggered message group rule.

&ZMSGTXTN

Contains the number of messages that triggered the message group rule.

&ZMSGTXT*n*

Contains the text of one of the messages that triggered the message group rule. *n* is a message arrival sequence number and is in the range 1 to &ZMSGTXTN.

Variables Not Available to EventView Correlation keys, Replacement Text, and ZREV Variables

The following variables are not available to EventView correlation keys, replacement text, and ZREV variables:

&ZMSGDATE

Contains the date of message arrival, in the form *dd-mmm-yyyy* (for example, 14-JUN-2000).

&ZMSGDAY

Contains the day of message arrival, in the form *ddd* (for example, WED).

Variables Not Available to EventView

The following variables are not available to EventView:

&ZMSGLINE*n*

Contains each line of a multi-line WTO or WTOR, where &ZMSGLINE1 is the major line.

&ZMSGLINES

Contains the number of lines in a multi-line WTO or WTOR.

&ZMSGLnW*n*

Contains a word from a line in a multi-line WTO or WTOR, for example, &ZMSGL24w16 is the 16th word in line 24.

Appendix J: Process Macros

This section contains the following topics:

Macros (see page 398)
AOMALERT Macro (see page 398)
CHAIN Macro (see page 398)
COMMAND Macro (see page 400)
COMPARE Macro (see page 403)
CONCAT Macro (see page 405)
DELALERT Macro (see page 406)
EVENT Macro (see page 406)
EVVARGET Macro (see page 409)
EVVARSET Macro (see page 410)
EXECNCL Macro (see page 411)
EXTRACT Macro (see page 412)
GENALERT Macro (see page 415)
GETSTAT Macro (see page 417)
GLBLSAVE Macro (see page 420)
GOTO Macro (see page 421)
PARSE Macro (see page 421)
PINGCD Macro (see page 425)
REPLY Macro (see page 426)
RUNPRCSS Macro (see page 429)
SETRC Macro (see page 430)
SETSTATE Macro (see page 430)
SETVARS Macro (see page 434)
SHDCALL Macro (see page 434)
SMFWRITE Macro (see page 436)
SNMPTRAP Macro (see page 437)
SOCKCLSE Macro (see page 438)
SOCKCONN Macro (see page 439)
SOCKRECV Macro (see page 440)
SOCKSEND Macro (see page 442)
STARTNCL Macro (see page 442)
STOP Macro (see page 444)
SUBJOB Macro (see page 445)
SUBPRCSS Macro (see page 448)
SVAPI Macro (see page 449)
SVCMD Macro (see page 450)
SYSCMD Macro (see page 451)
TRANS Macro (see page 455)
WAIT Macro (see page 457)
WAITEVNT Macro (see page 457)
WAITSTAT Macro (see page 460)
WTO Macro (see page 464)
WTOR Macro (see page 465)

Macros

Process macros are NCL procedures that are the building blocks of processes. Your product supplies a set of macros.

Each macro has a set of parameters, some of which have default settings which can be modified. Other parameters do not have a default setting and must be specified by you. You specify the parameters on the Macro Parameter Definition panel.

To display the Macro Parameter Definition panel, enter **P** beside the process step.

To determine which parameters (fields) on the Macro Parameter Definition screen are mandatory, press Enter.

The mandatory fields are highlighted.

Notes:

- Variables returned by a macro can be passed as parameters to the other macros in the same process.
- All macros set the return code in the &$RMMCRC variable. Unless specifically mentioned in the following descriptions, a zero return code means a successful operation and a nonzero return code means an unsuccessful operation.

AOMALERT Macro

Where possible, avoid using this macro. If you want to generate a WTO message, use the WTO macro.

The macro is based on the &AOMALERT NCL verb. For more information about the verb, see the *Network Control Language Reference Guide*.

CHAIN Macro

The CHAIN macro links the current process to another process that continues the execution.

You can chain processes together so that process A passes control to process B if certain conditions are met. You can define processes as modules and chain them as required. Use the CHAIN macro to chain processes. The macro sets the name of the target process and any associated parameters, and passes control to the target process.

Parameters: CHAIN Macro

Process

A mandatory field that specifies the name of the next process to be executed, provided that certain conditions are met. If no conditions are specified in the step containing the macro, the process executes automatically.

Parameters

Relevant if the process specified in the Process field contains variables that require you to pass values to them.

If the specified process does not require any passed values, this field remains empty.

Example: CHAIN Macro

In the following example, the PROC2 process is called and will be executed. PROC2 contains the variable &NAME which is given a value of TYOUNG.

```
PROD------ Automation Services : CHAIN Macro Parameter Definition -------------
Command ===>                                                    Function=UPDATE

.- Process Details ------------------------------------------------------------.
|                                                                              |
|  Process ..... PROC2                                                         |
|                                                                              |
|  Parameters .. NAME=TYOUNG                                                   |
|                _____________________________________________________________ |
|                                                                              |
'------------------------------------------------------------------------------'
```

In the following example, the PROC5 process is called and will be executed. PROC5 contains the variable &NAME which is given a value of &V1. The value of the &V1 variable can be set, for example, by an EventView message action rule.

```
PROD------ Automation Services : CHAIN Macro Parameter Definition -------------
Command ===>                                                    Function=UPDATE

.- Process Details ------------------------------------------------------------.
|                                                                              |
|  Process ..... PROC5                                                         |
|                                                                              |
|  Parameters .. NAME=&V1                                                      |
|                                                                              |
|                                                                              |
'------------------------------------------------------------------------------'
```

COMMAND Macro

The COMMAND macro issues a product command and interprets the results. It also lets you specify message text to detect the required response message.

Parameters: COMMAND Macro

Command

Is a mandatory field that specifies the name of the command to be issued by the macro.

This field can contain up to 61 characters.

Wait Time

Specifies the maximum time, in seconds, that the region waits for a response message. The time is reset each time a response is received. The macro waits until a message that matches a rule in the Message Text fields is received or until the time expires without receiving any more responses.

This field must contain a number from 1 through 9999. The default is 30.

Wait Time Expiry Return Code

Sets the return code if the expected message does not arrive before the expiry of the time specified in the Wait Time field.

This field must contain a number from 1 through 999. The default is 69.

Response Message Analysis

The following fields specify the rules that apply to response messages. You can specify up to five rules. The action codes enable you to refine the message text definition criteria. More information on the use of action codes is available in the online help.

Message Text

Specifies the initial part of the expected response message. It can contain up to 45 characters.

Return Code

Specifies the value returned by the macro when it receives the response message. The return code must be a number from 0 through 9999.

Extended Filter

Indicates whether an extended filter has been defined. This field is empty until the Message Text and the Return Code fields are completed. If you want to define an extended filter, enter **S** beside the rule to access the Extended Message Filtering panel.

Returned Variables: COMMAND Macro

&$RMMCMSGTEXT

Contains the full text of the message that satisfies the message rules.

&$RMMCWORD*n*

Contains the words in the message, where n indicates the position of the word in the message.

&ZRMMSG*n*

Contains any messages received before the required response message, where n is a number indicating the order in which those messages are received.

For example, if the response to a command consists of three messages A, B, and C, and message B satisfies the message rules, then:

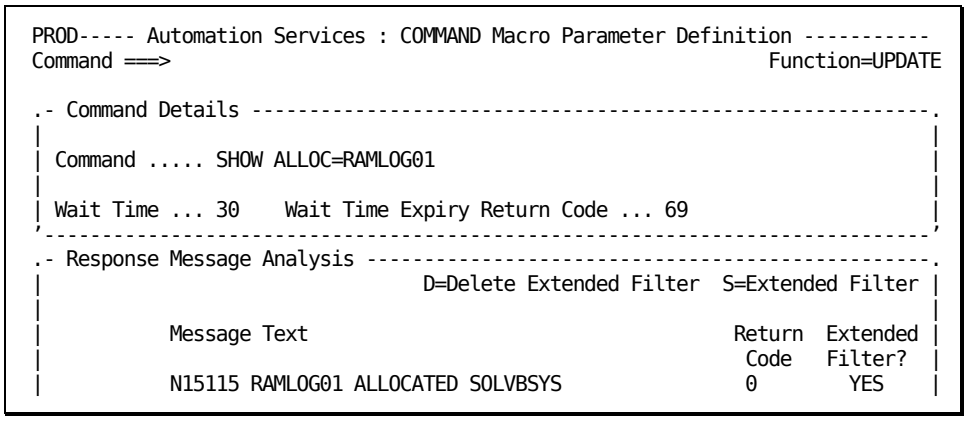
- &ZRMMSG1 contains message A
- &$RMMCMSGTEXT contains message B
- The macro ends on receipt of message B, and does not see message C

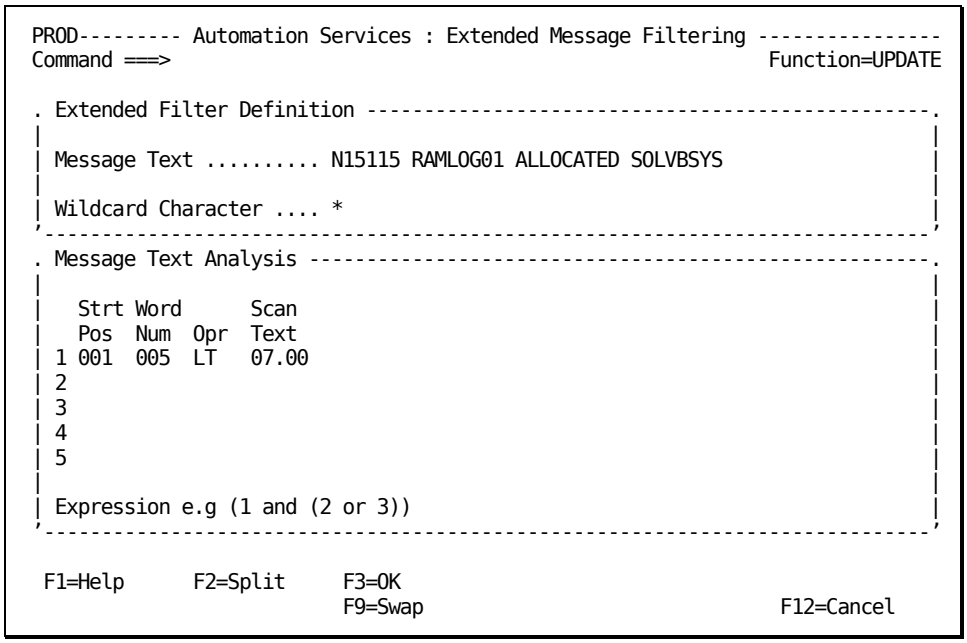
Example: COMMAND Macro

In the following example, the COMMAND macro executes the SHOW ALLOC=RAMLOG01 command and waits for a response message. In this case, that message is N15115 RAMLOG01 ALLOCATED SOLVBSYS *hh.mm*.

If this message is not received within 30 seconds, the return code is set to 69.

An extended filter checks for the value of *hh.mm*. More information about the completion of the Extended Message Filtering Panel is available from the online help.





COMPARE Macro

The COMPARE macro compares two values. You can:

- Compare the value in a variable with the value in another variable
- Compare the value in a variable with a fixed value
- Select a subset of the value to be compared

Parameters: COMPARE Macro

Variable Name 1

Is a mandatory field that is used to specify the variable that contains the value you want to compare. You can use the Start of Compare and the Length fields to compare a substring of the value.

Operator

Is a mandatory field that specifies the relational operator to use for the comparison. The value must be one of the following:

- CT (contains)
- EQ (is equal to)
- NE (is not equal to)
- GT (is greater than)
- GE (is greater than or equal to)
- LT (is less than)
- LE (is less than or equal to)

Variable Name 2

Is used to specify a variable that contains the value you want to use as the basis of the comparison.

Constant Value

Is used when you want to specify a fixed text string as the basis of the comparison.

Note: To use a null value for comparison, leave both the Variable Name 2 field and the Constant Value field blank.

Start of Compare

Indicates the start position of the string in variable 1 from which the comparison takes place. The value of this field must be a number from 1 through 255.

Length

Indicates how much of the string in variable 1 is to be compared. The value, if specified, must be a number from 1 through 255. If a value is not specified, the full string or substring, starting from the specified start position, is compared.

Return Codes: COMPARE Macro

0

Comparison criteria satisfied

4

Comparison criteria not satisfied

Example: COMPARE Macro

In the following example, the COMPARE macro checks whether the value of the &A3 variable is greater than the value of the &A2 variable.

```
PROD----- Automation Services : COMPARE MACRO Parameter Definition ------------
Command ===>                                                  Function=BROWSE

.- COMPARE Parameters ------------------------------------------------------.
|                                                                           |
|  Syntax .......... (Variable 1) OPERATOR (Variable 2)> or (Constant)      |
|                                                                           |
|                                                                           |
|  Variable Name 1 ....... A3            (Enter a valid variable name)      |
|                                                                           |
|  Operator .............. GT            (EQ,NE,GT,GE,LT,LE,CT)             |
|                                                                           |
|  Variable Name 2 ....... A2            (Enter a valid variable name)      |
|                                                                           |
|    or Constant Value ...                                                  |
|                                                                           |
|  Start of Compare ...... 1             (Start position within variable 1) |
|                                                                           |
|  Length .................              (Enter length of comparison)       |
|                                                                           |
'---------------------------------------------------------------------------'

 F1=Help      F2=Split     F3=Exit
                           F9=Swap
```

CONCAT Macro

The CONCAT macro concatenates specified data and stores the result in a variable.

Parameters: CONCAT Macro

Data

Specifies the data that you want to concatenate. The data can contain text and variables separated by spaces, which may either be preserved or be removed during concatenation depending on the value in the Preserve Blanks field. The result cannot be more than 256 characters. Excess characters are truncated.

Preserve Blanks

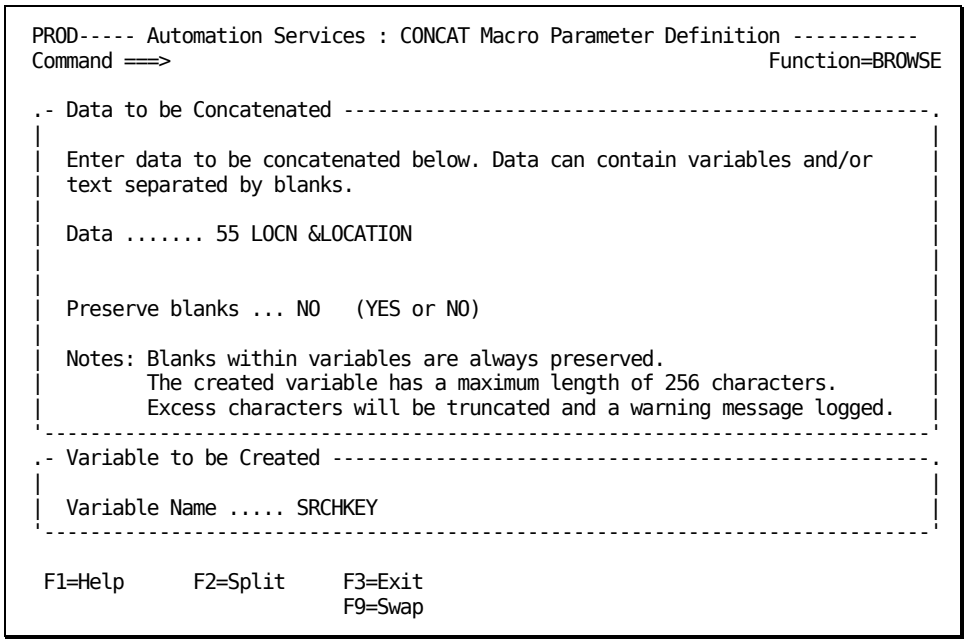
Specifies whether the spaces entered in the Data field should be preserved or removed during concatenation.

Variable Name

Specifies the name of the variable that contains the result of the concatenation.

Example: CONCAT Macro

In the following example, the CONCAT macro creates the &SRCHKEY variable, which contains the value 55LOCN*x*, where *x* is the value of the &LOCATION variable:



DELALERT Macro

The DELALERT macro deletes a user-generated alert.

Parameters: DEFALERT Macro

The DELALERT macro has only one parameter: the identifier of the alert to be deleted. The identifier is the alert reference key with which the alert was generated.

Example: DEFALERT Macro

In the following example, the DELALERT macro deletes an alert called UPSALERT9001:

```
PROD----------------- EventView : Alert Delete -------------------------------
Command ===>                                                  Function=BROWSE

. Alert Reference Key ------------------------------------------------------.
|                                                                           |
| Reference ... UPSALERT9001                                                |
```

EVENT Macro

The EVENT macro issues an N00102 event message that can be retrieved by NCL processes. To receive the message, the NCL process must be profiled for the event by using the PROFILE EDS command. For a description of the command, see the *Network Control Language Reference Guide*.

You can wait for the event message from within a process by using the WAITEVNT macro.

Parameters: EVENT Macro

Name

Identifies the event. The name can contain alphanumeric, @ and # characters, and . or _ characters (for example, SESSION_COMPLETION).

Type

Indicates the type of information contained in the event. Use the types as follows:

APPLICATION

To contain user-defined information.

ACCESS

To contain security alarm information.

CONFIGURATION

To contain object definition and relationship information.

PROCEDURAL

To contain scheduling and process control information.

SERVICEABILITY

To contain availability, degradation, error, fault, and recovery information.

UTILIZATION

To contain accounting, performance, response time, and statistics information.

Scope

Specifies the scope of delivery of the event. Valid values are as follows:

REGION

Restricts event delivery to processes in the region of the event issuer.

SYSTEM

Enables event delivery to processes in the entire region.

Object

Specifies a user-defined object class for the resource or resource pair for which the event is issued. If the value contains embedded spaces, the spaces are translated to underscore (_) characters.

Resource

Names the resource or resource pair (separated by a comma) for which the event is issued. An event can be issued for a pair of resources (for example, a session pair).

Reference

Specifies a code for the event.

Data

Specifies the data in the event.

Example: EVENT Macro

In the following example, the EVENT macro issues an OPERATIONS_CONDITION event. The event contains operations information in the &ZREVATTENDED user-defined EventView variable.

```
PROD------ Automation Services : EVENT Macro Parameter Definition -------------
Command ===>                                                    Function=BROWSE

.- Event Parameters ------------------------------------------------------------.
|                                                                               |
| Name........ OPERATIONS_CONDITION                                             |
|                                                                               |
| Type.......+ SERVICEABILITY                                                   |
| Scope......+ SYSTEM                                                           |
|                                                                               |
| Object......                                                                  |
| (Blanks will be translated to underscores)                                    |
|                                                                               |
| Resource....                                                                  |
| Reference...                                                                  |
|                                                                               |
| Data..... &ZREVATTENDED                                                       |
|                                                                               |
|                                                                               |
|                                                                               |
'-------------------------------------------------------------------------------'

 F1=Help      F2=Split     F3=Exit
                           F9=Swap
```

EVVARGET Macro

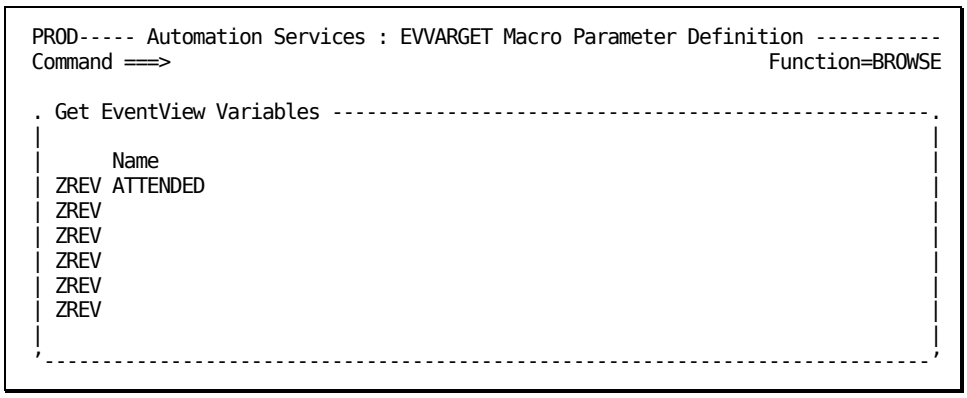
The EVVARGET macro retrieves the values of user-defined EventView variables. When obtained, these variables can be used in subsequent steps in the process that contains the macro.

Parameters: EVVARGET Macro

The EVVARGET macro has only one set of parameters; the character strings that, together with the ZREV prefix, form the names of the variables from which you want to retrieve the values. The character strings can contain variables that are resolved at execution time.

Example: EVVARGET Macro

In the following example, the EVVARGET macro retrieves the value of the &ZREVATTENDED variable. A message can then be sent describing the condition indicated by the value. The message can be used to modify operations behavior.



EVVARSET Macro

The EVVARSET macro sets the values of user-defined EventView variables.

Parameters: EVVARSET Macro

Name

Specifies the character string that, together with the ZREV prefix, forms the name of the variable.

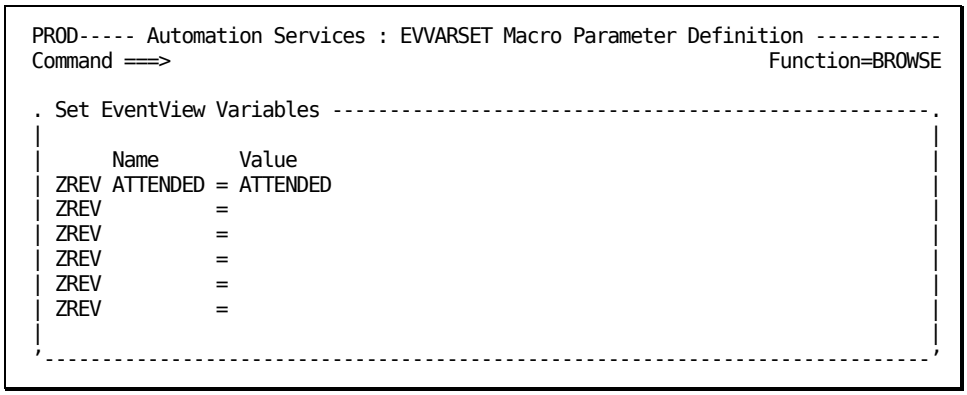
Value

Specifies the value of a named variable.

These fields can contain variables that are resolved at execution time. If you want to code other EventView variables here, they must be retrieved using EVVARGET in a previous step in the process.

Example: EVVARSET Macro

You want the EventView rules to react to messages differently depending on whether the system is attended, as indicated by the value of the &ZREVATTENDED variable. Use the WTOR macro to send a WTOR message and wait for the reply. If a reply is received within two minutes, set the &ZREVATTENDED variable to ATTENDED (see the following).



EXECNCL Macro

The EXECNCL macro executes an NCL procedure in line with the process that contains the macro. That is, the process waits for the NCL procedure to end before continuing on to the next step.

Parameters: EXECNCL Macro

NCL Name

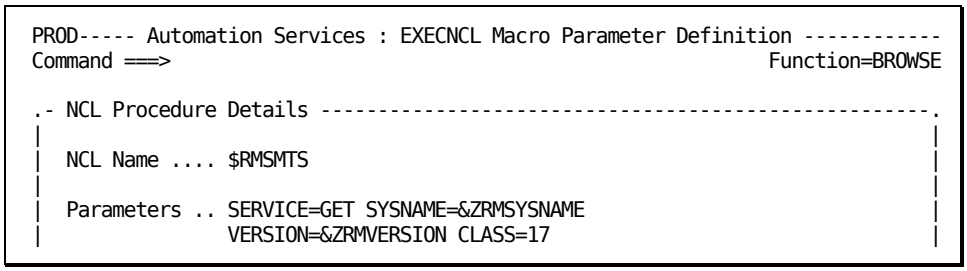
Specifies the name of the NCL procedure executed by the macro.

Parameters

Specifies any parameters required by the NCL procedure specified in the NCL Name field.

Example: EXECNCL Macro

In the following example, the EXECNCL macro executes the $RMSMTS procedure. The EXECNCL procedure contains a number of variables, which are defined in the Parameters field of this panel.



EXTRACT Macro

The EXTRACT macro extracts a segment from a text string.

Parameters: EXTRACT Macro

Where to Extract From—Input

Specifies the string from which you want to extract the text segment. This string can contain text and variables.

Variable to hold Extracted String—Output

Specifies the name of the variable in which you want to put the extracted text string.

How to do the Extraction

The following fields determine the text you want to extract. You must specify at least one of the Starting Position and Find Text fields. If you complete both fields, the macro looks first at the value of the Find Text field and then adds the value of the Starting Position field. The result locates the first character of the extracted segment.

Starting Position

Used in conjunction with the Length to Extract field, the Input field, and the Find Text field (if it is completed) to identify the text string you want to extract.

The Starting Position field accepts a number (*n*) from 1 through 256. The value specifies the first character of the text segment to be extracted.

Length to Extract

Specifies the number of characters you want to extract, effectively indicating the end point of the extracted text string.

The Length to Extract field accepts a number from 1 through 256. The value specifies the number of characters that will be in your extracted string. If a value is not specified in this field, the full string or substring is extracted.

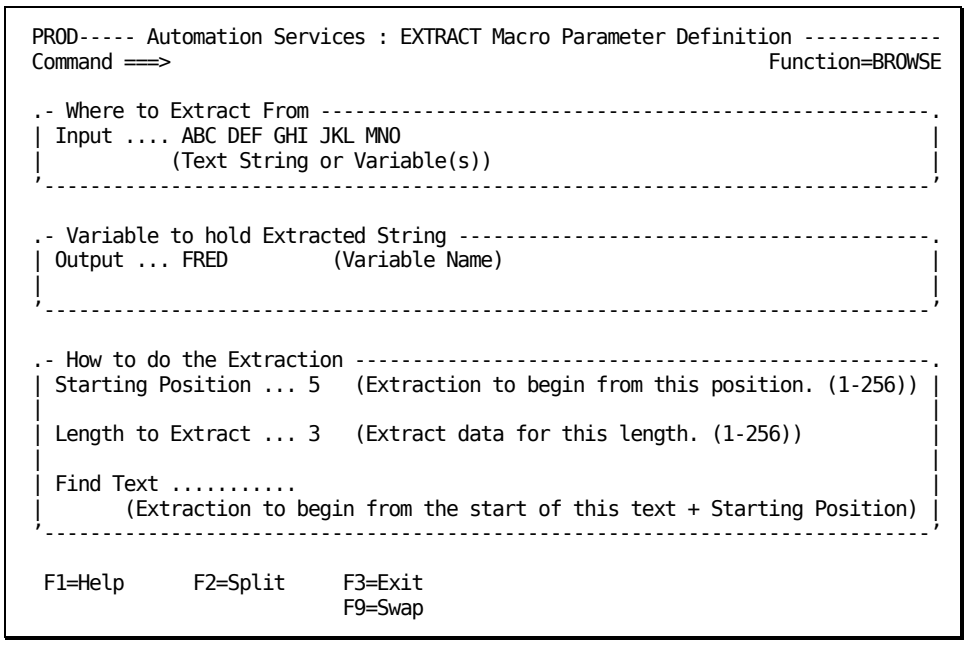
Find Text

If completed, must contain a section of the string specified in the Input field.

The Find Text field accepts a text string from 1 through 53 characters in length. The first character in this text string, plus the value in the Starting Position field, determines the first character of the extracted text. The last character in the extracted text is determined by the value in the Length to Extract field.

Example 1: EXTRACT Macro

In the following example, the EXTRACT macro is used to extract the text string DEF. Because no value is specified in the Find Text field, the reference point for the starting position is the first character of the value of the Input field. In this example, the first character of the extract is the 5th character of the Input text string (D), and because the length to extract is 3, the resulting extract is DEF.



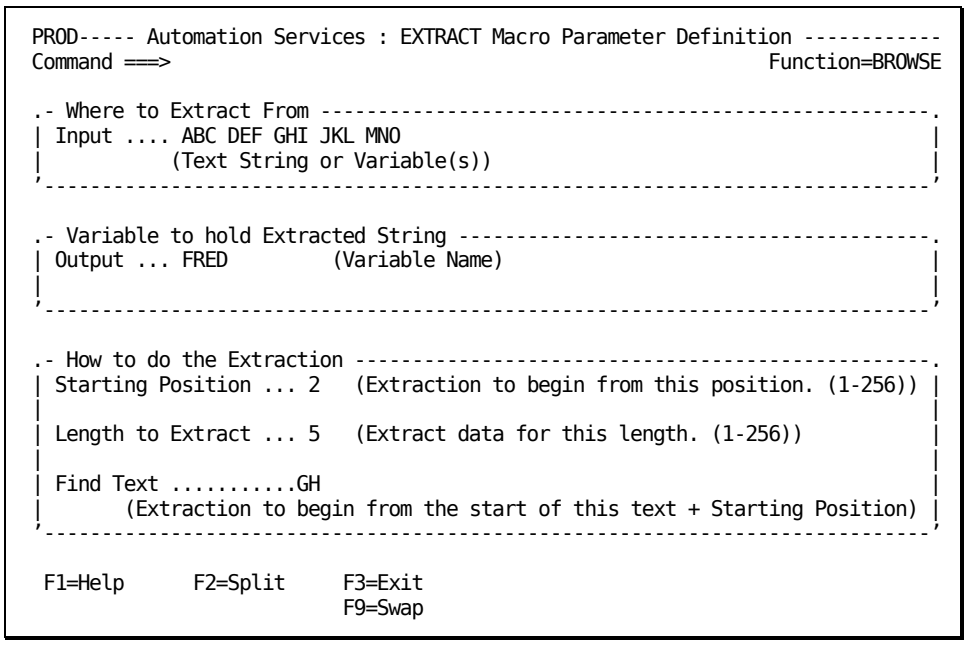
Example 2: EXTRACT Macro

In the following example, the EXTRACT macro is used to extract the text string GHI J. Because no value has been given to the Starting Position field, the reference point for the starting position is the value of the Find Text field. In this example, the first character of the extract is the first character of the Find Text string (G), and because the length to extract is 5, the resulting extract is GHI J.

```
PROD----- Automation Services : EXTRACT Macro Parameter Definition ------------
Command ===>                                                    Function=BROWSE

.- Where to Extract From ------------------------------------------------------.
| Input .... ABC DEF GHI JKL MNO                                               |
|           (Text String or Variable(s))                                       |
'------------------------------------------------------------------------------'

.- Variable to hold Extracted String ------------------------------------------.
| Output ... FRED         (Variable Name)                                      |
|                                                                              |
'------------------------------------------------------------------------------'

.- How to do the Extraction ---------------------------------------------------.
| Starting Position ...     (Extraction to begin from this position. (1-256)) |
|                                                                              |
| Length to Extract ... 5   (Extract data for this length. (1-256))            |
|                                                                              |
| Find Text ........... GH                                                     |
|       (Extraction to begin from the start of this text + Starting Position) |
'------------------------------------------------------------------------------'

 F1=Help      F2=Split     F3=Exit
                           F9=Swap
```

Example 3: EXTRACT Macro

In the following example, the EXTRACT macro is used to extract the text string HI JK. In this example, a value has been given to the Find Text field, and therefore that value becomes the reference point for the starting position. Because a value of 2 has been given to the Starting Position field, the first character of the extract is the second character of the Find Text string (H), and because the length to extract is 5, the resulting extract is HI JK.



GENALERT Macro

The GENALERT macro generates an alert for display on the alert monitor. The parameter definition comprises two panels. You must complete the Reference field on the Alert Attributes panel and the Alert Text field on the Alert Definition panel.

If you need to implement a process that generates variations of an alert, you can pass new GENALERT parameter values to the process to create the variations. The passed values override the values specified in the macro. You do not need to create a process for each variation of the alert.

You can specify what actions to take when the alert is raised. To specify actions, press F10 (Actions).

Parameters: GENALERT Macro

Reference

Identifies the alert.

To override this value, specify ALRTREF=*alert-reference* as a process parameter.

Severity

Specifies the importance of the alert. The valid values are 1 through 4. Severity 1 alerts are most important and severity 4 alerts are least important.

To override this value, specify ALRTSEV=*alert-severity* as a process parameter.

Origin

Specifies the source of the alert.

To override this value, specify ALRTORIGIN=*alert-origin* as a process parameter.

Alert Description

Specifies the text that describes the alert condition.

To override this value, specify ALRTDESC=*description* as a process parameter.

Alert Text

Provides a detailed description of the alert condition.

To override this value, specify ALRTTEXT*n*=*alert-text-n* (where *n* is 1 through 5, representing the five lines of text) as process parameters for the lines you want to override.

Alert Recommended Action

Specifies the actions that can be performed to remove the alert condition.

To override this value, specify ALRTACTN*n*=*action-line-n* (where *n* is 1 through 4, representing the four lines of the recommended action) as process parameters for the lines you want to override.

Example: GENALERT Macro

In the following example, the GENALERT macro generates the following UPSALERT9001 alert to remind the operator to test the uninterruptible power supply (UPS):

```
Test UPS
```

```
PROD----------------- EventView : Alert Attributes ---------------------------
Command ===>                                                  Function=BROWSE

. Alert Reference Key ------------------------------------------------------.
|                                                                            |
| Reference ... UPSALERT9001                                                 |
|                                                                            |
'----------------------------------------------------------------------------'
. Alert Attributes ---------------------------------------------------------.
|                                                                            |
| Severity .... 4                                                            |
| Origin ...... ALERTMACRO                                                   |
```

```
PROD----------------- EventView : Alert Definition ---------------------------
Command ===>                                                  Function=BROWSE

. Alert Description --------------------------------------------------------.
|                                                                            |
| TEST UPS                                                                   |
'----------------------------------------------------------------------------'
. Alert Text ---------------------------------------------------------------.
|                                                                            |
| UPS MAINTENANCE                                                            |
|                                                                            |
|                                                                            |
|                                                                            |
|                                                                            |
'----------------------------------------------------------------------------'
. Alert Recommended Action -------------------------------------------------.
|                                                                            |
| TEST THE UPS WHEN THERE ARE NO CRITICAL SYSTEM ACTIVITIES.                 |
|                                                                            |
|                                                                            |
|                                                                            |
'----------------------------------------------------------------------------'
 F1=Help      F2=Split     F3=Exit
 F7=Backward              F9=Swap     F10=Actions  F11=Panels
```

GETSTAT Macro

The GETSTAT macro discovers the status of a resource or service.

Parameters: GETSTAT Macro

System Name/Version

Identify the system image that owns the resource. If you do not specify an image, then during execution, the macro uses the following values:

- If the macro is associated with a resource definition, the macro uses the image that owns the definition.
- If the macro is not associated with a resource definition (for example, in an EventView rule), the macro uses the active local image.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Resource Class

Specifies the class name of the resource or service for which you require the status. This field can be completed manually, by entering the appropriate valid code, or automatically, by selecting a value from the prompted field value list. The list is displayed by entering **?** in the Resource Class field.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Resource Name

Specifies the name of the resource or service for which you want to retrieve the status.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Variable Prefix

Specifies the prefix of the variables that contain the retrieved status information. This prefix enables you to store the status information retrieved at different times in different variables.

The information is retrieved from the &ZRMST prefixed status variables. The retrieved information is stored in a corresponding set of variables, with the ZRMST prefix replaced by the specified prefix, *prefix*. For example, the information retrieved from the &ZRMSTACTSTAT variable is stored in the &*prefix*ACTSTAT variable.

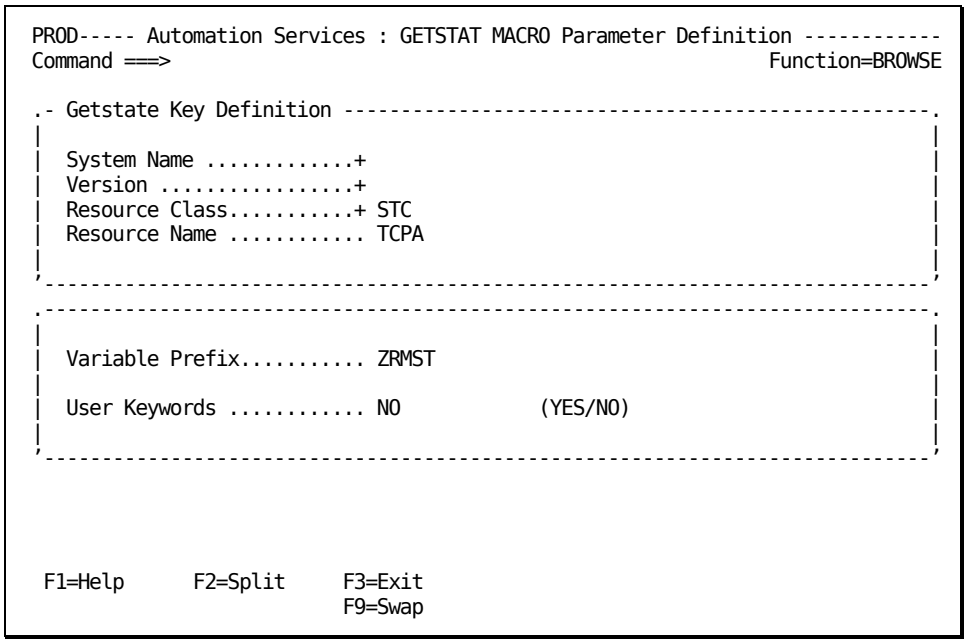
The prefix must *not* start with one of the following characters: # or $.

User Keywords

Enables you to store the keyword that is specified in the User Defined Keyword field on the Define Extended Display Attribute panel of a triggered message rule in a resource definition. The keyword is stored in the &*prefix*USRKEYW variable.

Example: GETSTAT Macro

In the following example, the GETSTAT macro retrieves the status of the STC class resource TCPA into variables of the form &ZRMST*.



GLBLSAVE Macro

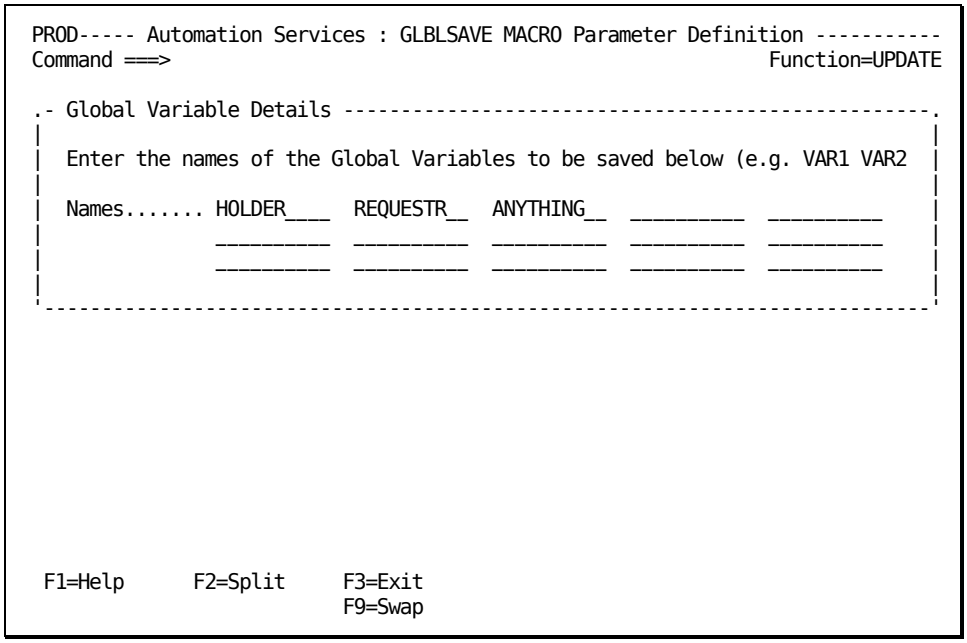
The GLBLSAVE macro saves the specified global variables so that their values are preserved when this region restarts.

Parameters: GLBLSAVE Macro

Names

Identifies the global variables you want to save. Each value is the name of a global variable without the global variable prefix.

Example: GLBLSAVE Macro



HOLDER

Identifies the global variable with the name *global_variable_prefix*HOLDER.

Note: The standard NCL *global_variable_prefix* is &&000.

GOTO Macro

The GOTO macro specifies the next process step to run.

Parameters: GOTO Macro

Target Step Label

Specifies the name that identifies the next process step to run.

Loop Control Limit

Specifies the runaway loop control limit. It prevents the process from being stuck in a loop.

Example: GOTO Macro

In the following example, the GOTO macro specifies that the STEP3 process step is to run next.

```
PROD------ Automation Services : GOTO Macro Parameter Definition --------------
Command ===>                                                    Function=UPDATE

. Goto Target ------------------------------------------------------------------.
|                                                                                |
|  Target Step Label .... STEP3___                                               |
|                                                                                |
|  Loop Control Limit ... 100                                                    |
|                                                                                |
'--------------------------------------------------------------------------------'
```

PARSE Macro

The PARSE macro segments a text string and puts the segmented contents in variables.

Parameters: PARSE Macro

String to be Segmented—Input

Specifies the string that you want to segment. This string can contain text and variables.

This is a mandatory field. The string can be from 1 through 64 characters in length.

Variables to be Created—Output

Specifies the name of the variables that are to hold your text segments.

This is a mandatory field that can hold from 1 through 64 characters. The names must be in one of the following formats:

varname

Use this format when you want to extract one segment only.

varname1,varname2,...,varnamen

Use this format when you want to extract more than one segment, when the segments are separated by delimiter characters such as a space.

varname1(length1),varname2(length2),...,varnamen(lengthn)

Use this format when you want to extract segments delimited by length.

prefix*

Use this format when you want to put extracted segments into variables &prefix1 through &prefix64.

If you want to skip a portion of the input string during parsing, specify *(*n*) where *n* represents the number of units to skip. An asterisk by itself is the same as *(1). Depending on the value in the By Length field, skipping occurs as follows:

- If the value is NO, *n* segments are skipped.
- If the value is YES, *n* characters are skipped.

How to Segment the Input String

The following fields determine how to segment the text string:

By Length

Indicates whether you want the segmentation to be by length.

This field accepts a YES or NO response. The default response is NO which indicates that the string is to be segmented first by delimiter and then by length. A YES response indicates that segmentation is to be by length only.

By Delimiter

Specifies the characters that are to be used as delimiters for text segments. You can specify up to 8 alphanumeric or special characters. The default character is a space.

Note: If you want to define a series of two characters as a delimiter, those characters must be enclosed in quotation marks.

Preserve Leading and Trailing Blanks?

Specifies whether you want leading and trailing blank spaces to be retained after segmentation. If this field is not completed, the default response of NO is assumed.

Create Null Variables if Consecutive Delimiters Found?

Specifies whether you want a variable (with a null value) to be created if consecutive delimiters are encountered. If this field is not completed, the default response of YES is assumed.

Remainder String Variable Name

Specifies the name of the variable that is to hold the text segment that remains after the specified parsing has been completed.

This field can contain a name from 1 through 12 characters in length.

Example: PARSE Macro

In the following example, the PARSE macro is used to segment the ABC 123 456 789 input string. The delimiter is the default (space). The segments are placed in the &FRW* variables.

The results will be:

- The variable &FRW1 contains the segment ABC
- The variable &FRW2 contains the segment 123
- The variable &FRW3 contains the segment 456
- The variable &FRW4 contains the segment 789

```
PROD------ Automation Services : PARSE Macro Parameter Definition -------------
Command ===>                                                  Function=BROWSE

.- String to be Segmented ------------------------------------------------------.
| Input .... ABC 123 456 789                                                     |
|            (Text String or Variable(s))                                       |
'--------------------------------------------------------------------------------'
.- Variables to be Created -----------------------------------------------------.
| Output ... FRW*                                                                |
|            (name or name1,name2... or name1(len),name2(len).. or prefix*)      |
'--------------------------------------------------------------------------------'
.- How to Segment the Input String ---------------------------------------------.
|    By Length ......          (YES or NO)                                       |
| OR By Delimiter ...          (Eight separate characters may be specified)      |
|                                                                                |
| Preserve Leading and Trailing Blanks? ..................... NO  (YES or NO)  |
|                                                                                |
| Create NULL Variables if consecutive delimiters found? .... YES (YES or NO)  |
|                                                                                |
| Remainder String Variable Name (If remainder required) ....                    |
'--------------------------------------------------------------------------------'

 F1=Help      F2=Split     F3=Exit
                           F9=Swap
```

PINGCD Macro

The PINGCD macro checks the availability of a destination (remote) node for a CONNECT:Direct region.

Parameters

CDMGR Name

Specifies the name of the manager for the CONNECT:Direct region to which the destination node is defined.

Remote Node

Identifies the destination node whose availability is to be checked.

Wait Time

Specifies the time (in seconds) to wait for a response.

If the macro does not receive a response by the specified time, a user-specified return code is set.

Limits: 0 to 9999

Wait Time Expiry Return Code

Specifies the code to set when the macro does not receive a response by the time specified in the Wait Time field.

Limits: 0 to 999

Returned Codes

0

Indicates that the destination node is available.

4

Indicates that the destination node cannot be contacted.

8

Indicates that the destination node is not defined in the network map of the specified CONNECT:Direct region.

REPLY Macro

The REPLY macro replies to an outstanding write-to-operator with reply (WTOR) message. The reply can be to one of the following:

- The last WTOR message
- The first WTOR message from a particular job
- A specified WTOR message

The REPLY macro lets you specify message rules to detect any expected response messages.

Parameters: REPLY Macro

Last WTOR?

Enter YES in the Last WTOR? field to indicate that the message in the Reply Text field is in response to the last WTOR message seen by this process (for example, a WTOR message sent by a previous step). If the message is not intended as a reply to the last WTOR, leave the field blank.

Note: If your reply is intended for the last WTOR message, complete the Last WTOR field only. Do not complete either the Jobname or WTOR Message Text fields.

Jobname

If the message in the Reply Text field is intended as a reply to a WTOR message from a particular job, enter the name of the job in the Jobname field.

If you also complete the WTOR Message Text field, your reply is to that message received from the named job. If you do not complete the WTOR Message text field, your reply is to the first WTOR message received from the named job.

Note: If you complete this field, you cannot also complete the Last WTOR? field.

WTOR Message Text

Specifies the text of the message to which you want to reply. It need not contain the complete message, but must contain sufficient text to adequately identify the message.

If you also complete the Jobname field, your reply is to the WTOR message from the named job only.

If you complete this field, you cannot also complete the Last WTOR? field.

Reply

Specifies your reply.

Limits: 1 through 65 characters in length.

Message Text

Specifies the initial part of the expected response message.

Limits: Up to 45 characters.

R/Code

Specifies the value to be returned by the macro when it receives the response message.

Limits: A number from 0 through 999.

E/Filter

Displays YES or NO indicating whether an extended filter has been defined. This field is completed automatically when a rule is defined. It cannot be modified.

Wait Time

Specifies the maximum time, in seconds, that the region waits for a response message. The time is reset each time a response is received. The macro waits until a message that matches a rule in the Message Text fields is received or until the time expires without receiving any more responses.

Limits: A number from 1 through 9999.

Default: 30

Wait Time Expiry Return Code

Sets the return code if the expected message does not arrive before the expiry of the time specified in the Wait Time field.

Limits: A number from 1 through 999.

Default: 69

Returned Variables: REPLY Macro

&$RMMCMSGTEXT

Contains the full text of the message that satisfies the message rules.

&$RMMCWORD*n*

Contain the words in the message, where n indicates the position of the word in the message.

&ZRMMSG*n*

Contain any messages received before the required response message, where n is a number indicating the order in which those messages are received.

&$RMMCWTORID

Contains the ID of the last encountered WTOR message if the macro encounters any WTOR messages before it encounters an expected message.

For example, if the response to a command consists of three messages A, B, and C, and message B satisfies the message rules, then:

- &ZRMMSG1 contains message A
- &$RMMCMSGTEXT contains message B
- The macro ends on receipt of message B, and does not see message C

Example: REPLY Macro

In the following example, the REPLY macro is used to reply to the WTOR message identified by the following text string:

```
T00IJ013R CONFIRM REQUEST TO STOP A/S
```

```
PROD------ Automation Services : REPLY Macro Parameter Definition -------------
Command ===>                                                  Function=UPDATE

.- WTOR Identification ------------------------------------------------------.
| Last WTOR? Or Jobname And/Or WTOR Message Text                              |
|    ___ (Yes)  ________       T00IJ013R CONFIRM REQUEST TO STOP A/S_________ |
'-----------------------------------------------------------------------------'
.- Reply Text ---------------------------------------------------------------.
|                                                                             |
| Reply.... Y_____________________________________________________________    |
'-----------------------------------------------------------------------------'
.- Response Message Analysis ------------------------------------------------.
|                                D=Delete Extended Filter  S=Extended Filter  |
|           Message Text                                    R/Code  E/Filter  |
|   ___     T00IJ014I_____________________________________     _0_            |
|   ___     ______________________________________________     ___            |
|   ___     ______________________________________________     ___            |
|   ___     ______________________________________________     ___            |
|   ___     ______________________________________________     ___            |
|                                                                             |
| Wait Time 30__  Wait Time Expiry Return Code 69_                            |
```

RUNPRCSS Macro

The RUNPRCSS macro runs another process in line with the current process. The current process waits for the other process to end before continuing to the next step.

Parameters: RUNPRCSS Macro

Process

Specifies the name of the process to be executed by the macro.

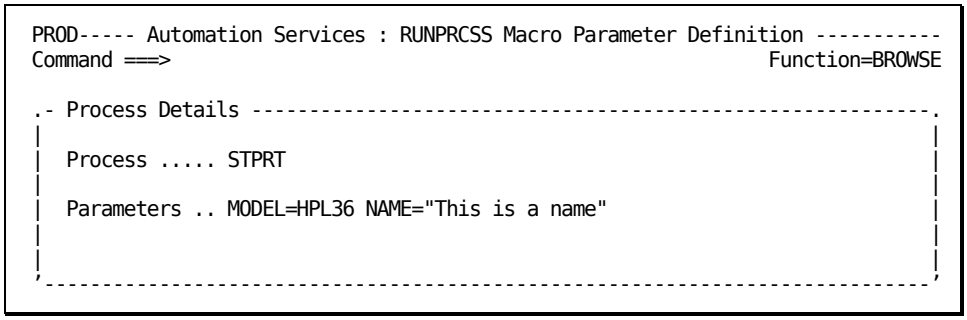
Parameters

Only relevant if the process specified in the Process field contains variables that require you to pass values to them.

If the specified process does not require any passed values, leave this field empty.

Example: RUNPRCSS Macro

In the following example, the RUNPRCSS macro runs the STPRT process. The STPRT process contains the &MODEL and &NAME variables that are defined in the Parameters field of this panel.



SETRC Macro

The SETRC macro sets the process return code.

Parameters: SETRC Macro

Return Code

Specifies the value that is returned by a macro when each process step is completed. This is the trigger for the next step to begin, provided that any specified conditions are met.

Limits: A number from 0 through 99

Example: SETRC Macro

In the following example, the SETRC macro is used to set the return code to 21.

```
PROD------ Automation Services : SETRC Macro Parameter Definition -------------
Command ===>                                                    Function=BROWSE

.- Return Code ------------------------------------------------------------------.
|                                                                                 |
|  Return Code .... 21  (Enter a number between 0 and 99)                         |
|                                                                                 |
'---------------------------------------------------------------------------------'
```

SETSTATE Macro

The SETSTATE macro sets the actual state and the extended display attributes of a resource or service.

Parameters: SETSTATE Macro

System Name/Version

Identifies the system image that owns the resource. If you do not specify an image, then during execution, the macro uses the following values:

If the macro is associated with a resource definition, the macro uses the image that owns the definition.

If the macro is not associated with a resource definition (for example, in an EventView rule), the macro uses the active local image.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Resource Name

Specifies the resource or service for which you are setting the state. The value of this field can be a variable whose value is passed to this macro by the process itself.

This field is mandatory.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Resource Class

Specifies the class of the item named in the Resource Name field.

Entering a question mark (?) in the Resource Class field displays the Resource Class List panel from which you can select the appropriate class.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

State

Specifies the actual state you want to set for the named resource or service.

This field is mandatory and must contain one of the following (entered either directly or by using a variable):

- ACTIVE
- DEGRADED
- FAILED
- INACTIVE
- STARTING
- STOPPING
- UNKNOWN

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Description

Specifies the free form text that can be used to identify who has set the state or the reason for setting that state. The value in this field is displayed as part of the detailed status information, which is accessed from a monitor.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Intensity

Specifies the intensity of the display text and must be either HIGH or LOW. If no value is set, the default value of LOW is assumed.

Color

Specifies the color of the display text and must be one of BLUE, GREEN, PINK, RED, TURQUOISE, WHITE, or YELLOW. If no value is set, the default value of GREEN is assumed.

Highlight

Specifies the highlight used for the display text and must be one of BLINK, REVERSE, or NONE. If no value is set, the default value of NONE is assumed.

Icon Flag

Specifies whether the extended display color of a resource that is in the worst logical state should be used on an icon.

If you do not want the icon to use extended display color, specify NO (the default) in this field. For example, if the resource does not require attention, transferring the color to the icon can cause an undesirable change in the icon color.

If you want the icon to use extended display color, specify YES in this field. This setting also forces the state ranking of the resource to equal that of the FAILED logical state. The icon can thus be forced to display this resource even though it is not in the worst logical state. For example, if you rate the condition of a tape mount request very important, specify YES in the Use on Graphic Monitor? field.

Text

Specifies the free form extended status text that overlays the current display.

If you do not want to change the status display, leave the field blank.

If you want to reset the status line to normal display, type ##RESET## in this field.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Example: SETSTATE Macro

In the following example, the SETSTATE macro is used to set the actual state and the extended display according to the parameters passed to the process that contains the macro.

```
PROD----- Automation Services : SETSTATE Macro Parameter Definition -----------
Command ===>                                                  Function=BROWSE

.- Resource Identification ------------------------------------------------.
| System Name ......+                                                      |
| Version ..........+                                                      |
| Resource Class ...+ &CLS                                                 |
| Resource Name ..... &NAME                                                |
'--------------------------------------------------------------------------'
.- Actual State -----------------------------------------------------------.
| State ............ &V1      (Active, Inactive, Failed, Degraded,          |
|                             Starting, Stopping or Unknown)               |
| Description ...... &REASON                                               |
'--------------------------------------------------------------------------'
.- Extended Display -------------------------------------------------------.
| Intensity ... LOW        (High or Low)                                   |
| Color ....... GREEN      (Blue, Green, Pink, Red, Turquoise, White, Yellow) |
| Highlight ... NONE       (Blink, Reverse or None)                        |
| Icon Flag ...            (Use Attributes to change Icon color - Yes or No)  |
|                                                                          |
| Text ........ &EXTDISP                                                   |
'--------------------------------------------------------------------------'
 F1=Help      F2=Split     F3=Exit
                           F9=Swap
```

SETVARS Macro

The SETVAR macro creates variables for a process.

The SETVARS macro stores the values of transient variables. For example, the values of the &$RMMCWORD*n* variables change for each executed COMMAND, REPLY, and SYSCMD macro. Using the SETVARS macro, you can store the original values before they change.

Parameters: SETVARS Macro

Variable String

Defines the variables (and their values) required by the process. These variables are passed to all macros in the process.

Example: SETVARS Macro

In the following example, the SETVARS macro sets the value of the &V1 and &V2 variables.

```
PROD----- Automation Services : SETVARS Macro Parameter Definition ------------
Command ===>                                                    Function=BROWSE

.- Variable Details ------------------------------------------------------------.
|                                                                               |
|  Enter the variables to be created below. (e.g. VAR1=TEST VAR2=&$RMMCWORD1)   |
|                                                                               |
|  Variable String... V1=ACTUAL V2=INACTIVE                                     |
|                                                                               |
|                                                                               |
|                                                                               |
'-------------------------------------------------------------------------------'
```

SHDCALL Macro

The SHDCALL macro passes a command to the CA 7 batch scheduler and returns the response.

Note: If you want to use this macro, you must first complete and action the SCHEDAPI Customizer parameter group (shortcut **/PARMS**).

Parameters: SHDCALL Macro

Scheduler Product

Specifies the scheduler product to which the commands are passed.

Scheduler SSID

Specifies the subsystem ID of the scheduler product to which the commands are passed. The default is the value entered in the SCHEDAPI Customizer parameter group.

Command

Specifies one or more commands passed to the scheduler product. Use a semicolon (;) as a command separator.

Time-out

Specifies the maximum time, in seconds, that the region waits for a response from the scheduler product. The time is reset each time a response is received. The macro waits until a message is received, or until the time expires without receiving any response.

Limits: A number from 1 through 999.

Default: 30.

Returned Variables: SHDCALL Macro

&RMCNT

The number of response text lines.

&RM1 to &RM999

Every line in the response.

&RMMSG

The error message if the return code is not zero.

&RMRC

The return code

To run the macro interface, the STEPLIB statement in the CA 7 started task JCL must be concatenated in the CA NetMaster started tasks JCL. Following is an example:

```
//STEPLIB DD DSN=dsnpref.pvpref.MS.CC2DLOAD,
//        DD DISP=SHR,DSN=CD.Vnnnn.LINKLIB
//        DD DISP=SHR,DSN=dsnpref.CA7Vnn.CAILIB
//        DD DSN=SYS1.SISTCLIB,DISP=SHR
```

SMFWRITE Macro

The SMFWRITE macro writes a user-defined SMF record (SMS record on VOS3 systems).

Parameters: SMFWRITE Macro

Test Mode

Can be used to test the macro.

If the test mode is OFF, the macro writes the specified record to the SMF (or SMS) data set. If the test mode is ON, the macro sends the specified record to the activity log.

Subtype

Specifies the record subtype. The 9*xxx* subtypes are reserved for these user-defined records.

Enter a hexadecimal value. The subtype is constructed by prefixing the specified value with 9.

SMF Data

Specifies the data in each field of the record. This field may contain constant or variable data (data contained in variables).

Length

Specifies the length of a data field in bytes.

Example: SMFWRITE Macro

In the following example, the SMFWRITE macro writes a record that indicates when a service starts.

```
--------- Automation Services : SMFWRITE Macro Parameter Definition -----------
Command ===>                                                  Function=BROWSE

.- SMF Parameters ------------------------------------------------------------.
|                                                                             |
| Test Mode ... OFF                                                           |
| Subtype ..... FFF                                                           |
|                                                                             |
|   SMF Data                                                        Length    |
|   START                                                             6       |
|   &SERVICE                                                          12      |
```

SNMPTRAP Macro

The SNMPTRAP macro generates an SNMP trap. You can use an SNMP trap to inform a remote system of a resource or service state change.

Parameters: SNMPTRAP Macro

Text

Specifies the text to send with the trap. The format of the text should conform to the specified enterprise ID and specific trap number.

Destinations Dataset

Specifies the data set name (DSN) in which the addresses of remote systems are located.

Destination Address(es)

Specifies the destination addresses of the remote systems. You can enter an address, or multiple addresses by using a comma or a space as the delimiting character.

Enterprise ID

Together with the specific trap number, specifies the format of the trap. The format identified by this field is determined by your open platform administration.

Specific Trap Number

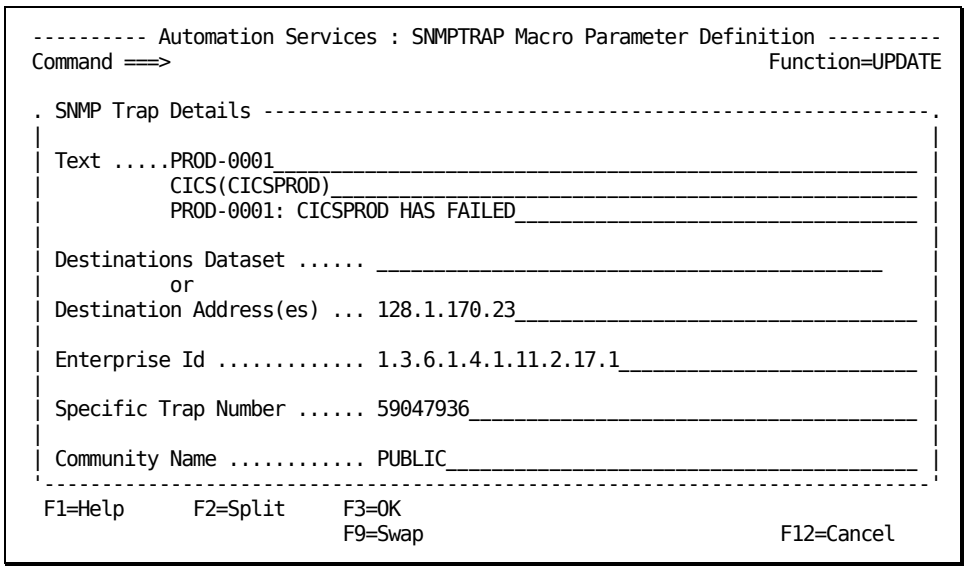
Together with the enterprise ID, specify the format of the trap. The format identified by this field is determined by your open platform administration.

Community Name

Specify the community for which the trap is destined. This field is **case sensitive** and can be left with its default of public.

Example: SNMPTRAP Macro

In the following example, the SNMPTRAP macro sends a trap to the remote system at the specified address when the CICS task is lost. The macro is specified in a process definition that is used by the resource definition for the task (CICSPROD) and processed as an exit upon receipt of a message which indicated that the CICS region has failed.



SOCKCLSE Macro

The SOCKCLSE macro closes the socket connection, which was opened using the SOCKCONN macro.

Parameters: SOCKCLSE Macro

The SOCKCLSE macro does not have any parameters. There can be one open connection only and this is the one that is closed.

SOCKCONN Macro

The SOCKCONN macro establishes a socket connection with an IP address. It is used to determine the availability of a connection.

Parameters: SOCKCONN Macro

IP Address/Host Name

Specifies the IP address or host name of the application to which you want to connect.

Port Number

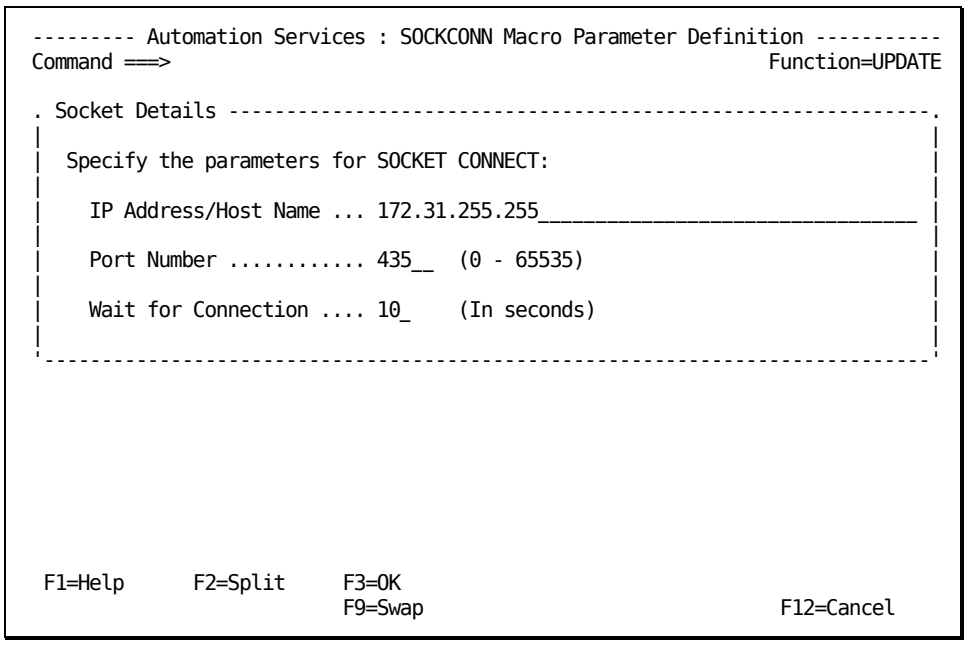
Specifies the port number of the application to which you want to connect.

Wait for Connection

Specifies the time (in seconds) that you want to try to establish the connection.

Example: SOCKCONN Macro

In the following example, the SOCKCONN macro is used to connect to port number 435 at IP address 172.31.255.255. The system will wait 10 seconds for the connection.



SOCKRECV Macro

The SOCKRECV macro receives data (via the established socket connection) from an IP application. It is used to determine the health of an application. You can specify up to three different data streams, which can then set three different return codes.

Parameters: SOCKRECV Macro

Expected Data Format

Specifies the format in which you want to receive the data. You can choose either ASCII or EBCDIC.

Wait Time

Specifies the time (in seconds) that you want to wait to receive the data.

RC

Specifies the return code for the data.

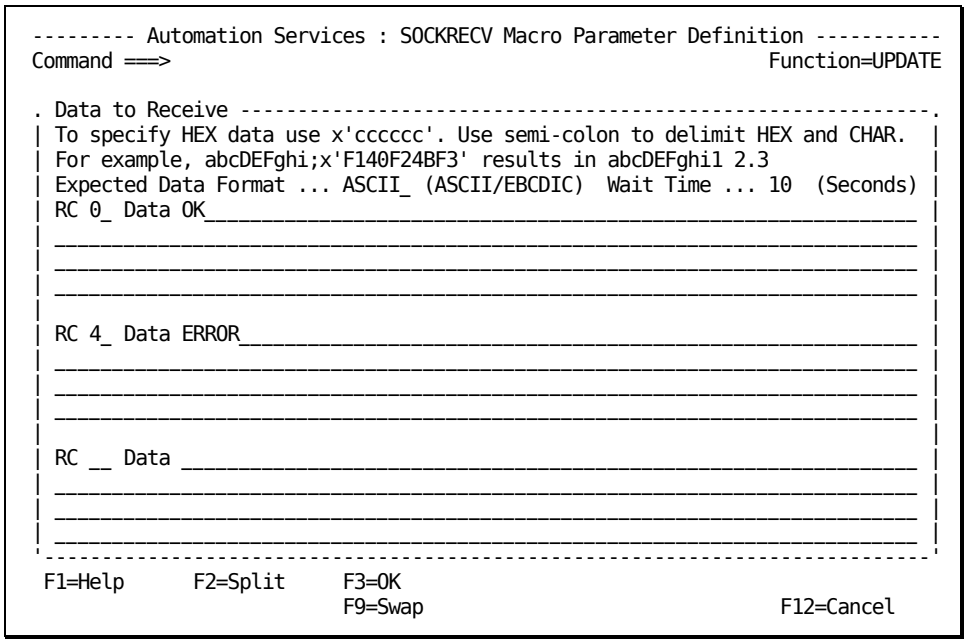
Data

Specifies the data that you want returned.

Unprintable characters, for example hexadecimal, can be entered by preceding the command with **x'** and ending it with **'**. For example, x'456703'. The data must also be separated from text with a semicolon (;).

Example: SOCKRECV Macro

In the following example, the SOCKRECV macro receives data from the open socket connection. If OK is received, a code of 0 is returned; if ERROR is received, a code of 4 is returned. The data is returned in ASCII format and the system waits 10 seconds for a response.



SOCKSEND Macro

The SOCKSEND macro sends data (via the established socket connection) to an IP application. It is used to determine the health of an application.

Parameters: SOCKSEND Macro

Data Format

Specifies the format in which you want to send the data. You can choose either ASCII or EBCDIC.

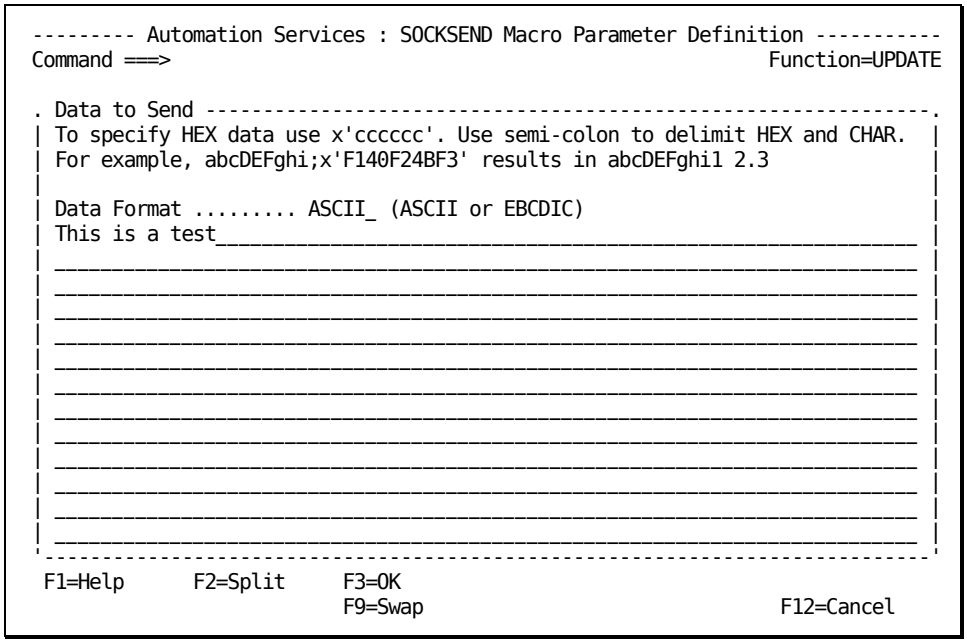
Data

Specifies the data you want to send.

Unprintable characters, for example hexadecimal, can be entered by preceding the command with **x'** and ending it with **'**. For example, x'456703'. The data must also be separated from text with a semicolon (;).

Example: SOCKSEND Macro

In the following example, the SOCKSEND macro sends the data 'This is a test' in ASCII format to the IP address with the open socket connection.



STARTNCL Macro

The STARTNCL macro starts an NCL procedure that runs independently of the process.

Parameters: STARTNCL Macro

NCL Name

Specifies the name of the NCL procedure you want to start.

Region

Identifies the region in which the specified NCL procedure is to start. The field must be blank or contain one of the following background environments:

- AOMP
- BLOG
- BMON
- BSYS
- CNMP
- LOGP
- PPOP

If this field is blank, the current region is assumed.

Parameters

Specifies the parameters (and their values) required by the NCL procedure specified in the NCL name field.

Example: STARTNCL Macro

In the following example, the STARTNCL macro starts the CALC2 NCL procedure which contains the &MULT variable. The procedure is started in the current region.

```
PROD----- Automation Services : STARTNCL Macro Parameter Definition -----------
Command ===>                                                  Function=BROWSE

.- NCL Procedure Details ------------------------------------------------------.
|                                                                              |
|  NCL Name .... CALC2                                                         |
|                                                                              |
|  Region .....+           (AOMP, BLOG, BMON, BSYS, CNMP, LOGP or PPOP)        |
|                                                                              |
|  Parameters .. MULT=365                                                      |
|                                                                              |
|                                                                              |
|                                                                              |
|                                                                              |
'------------------------------------------------------------------------------'
```

STOP Macro

The STOP macro stops a process after setting the return code.

Parameters: STOP Macro

Return Code

Lets you set the process return code.

Limits: A numeric value from 0 through 99.

Example: STOP Macro

In the following example, the return code for the STOP macro is set to 0.

```
PROD------ Automation Services : STOP Macro Parameter Definition --------------
Command ===>                                                    Function=BROWSE

.- Return Code ------------------------------------------------------------.
|                                                                           |
|  Return Code .... 0   (Enter a number between 0 and 99)                   |
|                                                                           |
'---------------------------------------------------------------------------'
```

SUBJOB Macro

The SUBJOB macro submits a JCL batch job.

Parameters: SUBJOB Macro

Dataset Name

Specifies a mandatory field that specifies the data set member that contains the job steps.

This field can contain a sequential data set.

Wait Time

Specifies the maximum time, in seconds, that the region waits for a response message. The time is reset each time a response is received. The macro waits until a message that matches a rule in the Message Text fields is received or until the time expires without receiving any more responses.

Limits: 1 to 9999.

Default: 30

Wait Time Expiry Return Code

Specifies the value of the macro return code if the expected message does not arrive before the expiry of the time specified in the Wait Time field.

Limits: 1 to 999.

Default: 69

Message Text

Specifies the initial part of the required response message. The text can contain up to 45 characters.

Return Code

Specifies the value set by the macro when it receives the response message.

Limits: 0 to 999.

Extended Filter?

Indicates whether an extended filter has been defined. This field is empty until you complete the Message Text and Return Code fields. If you want to define an extended filter, enter **S** to access the Extended Message Filtering panel.

After this panel is completed, the Extended Filter field is updated automatically.

Returned Variables: SUBJOB Macro

&$RMMCMSGTEXT

Contains the full text of the message that satisfies the message rules.

&$RMMCWORD*n*

Contains the words in the message, where *n* indicates the position of the word in the message.

&ZRMMSG*n*

Contains any messages received before the required response message, where *n* is a number indicating the order in which those messages are received.

&$RMMCWTORID

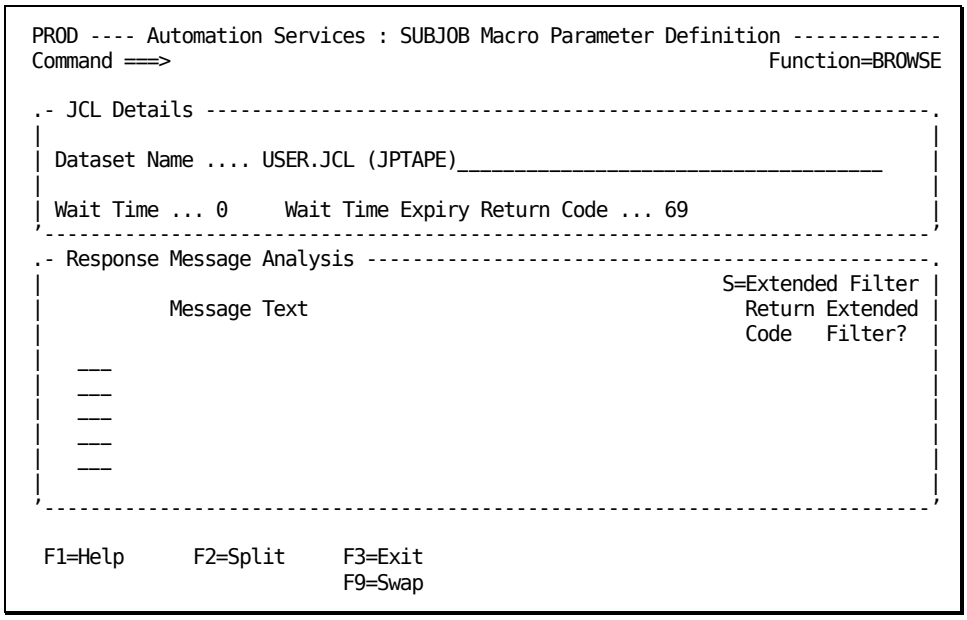
Contains the ID of the last encountered WTOR message if the macro encounters any WTOR messages before it encounters an expected message.

For example, if the response to a command consists of three messages A, B, and C, and message B satisfies the message rules, then:

- &ZRMMSG1 contains message A
- &$RMMCMSGTEXT contains message B
- The macro ends on receipt of message B and does not see message C

Example: SUBJOB Macro

In the following example, the name of the data set member that contains the job steps is USER.JCL (JPTAPE).



SUBPRCSS Macro

The SUBPRCSS macro submits another process to a region. The process runs separately from the current process—that is, *not* in line with the current process.

Parameters: SUBPRCSS Macro

Process

Specifies the name of the process you want to start.

Region

Identifies the owner of the processing region in which the process is to start. In this case, the owner is a background user and must be one of:

- AOMP
- BLOG
- BMON
- BSYS
- CNMP
- LOGP
- PPOP

This is a mandatory field.

Log

Indicates the way in which you want the process activity logged to the activity log. The value of this field must be one of the following:

- NO (no log)
- FULL (full log)
- SUMM (summary log)
- BOTH (both full and summary logs)

If this field is not completed, the default value of NO is assumed.

Parameters

Specifies the parameters (and their values) required by the specified process.

Example: SUBPRCSS Macro

In the following example, the process STAMP runs in the BLOG environment. The activities of STAMP are fully logged. STAMP contains the &NAME variable, which is defined as WTOH.

```
PROD----- Automation Services : SUBPRCSS Macro Parameter Definition -----------
Command ===>                                                    Function=BROWSE

.- Process Details ------------------------------------------------------------.
|                                                                              |
|  Process ..... STAMP                                                         |
|                                                                              |
|  Region .....+ BLOG      (AOMP, BLOG, BMON, BSYS, CNMP, LOGP or PPOP)        |
|                                                                              |
|  Log ........+ FULL      (NO, FULL, SUMM, BOTH)                              |
|                                                                              |
|  Parameters .. NAME=WTOH                                                     |
|                                                                              |
|                                                                              |
'------------------------------------------------------------------------------'
```

SVAPI Macro

The SVAPI macro executes the $RMCALL API. The macro inherits the $RMCALL return code.

Parameters: SVAPI Macro

Opt

Refers to the requested option and is automatically set to SERVICE.

Service

Refers to the requested SERVICE and is automatically set to ACTION.

Action

Mandatory field that identifies the type of processing required. The value of this field must be one of the following:

- COMMAND issues an Automation Services command.
- DBGET retrieves the definition of a resource or service specified by the SysName, Version, Class, and Name fields. The information is stored in the &ZRMDB prefixed variables.
- STGET retrieves the current status of the resource or service specified by the SysName, Version, Class and Name fields. The information is stored in the &ZRMST prefixed variables.

Command

Identifies the required command if the value of the Action field is COMMAND.

SysName

Identifies the system image in which to perform the action. If this field is not completed, the current system image is assumed by default.

Version

Identifies the version of the system image in which to perform the action. If this field is not completed, the current system image is assumed by default.

Class

Identifies the resource or service class.

Name

Identifies the resource or service.

Parms

Specifies the parameters required by the command specified in the Command field.

Example: SVAPI Macro

In the following example, the macro sets the actual state of the local CICSA resource to ACTIVE.

```
PROD------ Automation Services : SVAPI Macro Parameter Definition -------------
Command ===>                                                  Function=UPDATE

.- API Parameters ------------------------------------------------------------.
|                                                                             |
|  Opt ........ SERVICE             Service .... ACTION                       |
|                                                                             |
|  Action ..... COMMAND             Command .... ASA                          |
|                                                                             |
|  SysName ....                     Version ....                              |
|                                                                             |
|  Class ...... 02                  Name ....... CICSA                        |
|                                                                             |
|  Parms ......                                                               |
|                                                                             |
'-----------------------------------------------------------------------------'



 F1=Help      F2=Split     F3=OK
                           F9=Swap                                F12=Cancel
```

SVCMD Macro

The SVCMD macro issues a registered Automation Services command. The macro inherits the $RMCALL return code.

Parameters: SVCMD Macro

Command Name

A mandatory field that specifies the name of the registered command issued by the SVCMD macro.

To display a full list of registered commands, enter **?**.

Parameters

Specifies the parameters passed to the specified command.

Example: SVCMD Macro

The following example issues the T (Terminate) command for the local resource named in &NAME.

```
PROD------ Automation Services : SVCMD Macro Parameter Definition -------------
Command ===>                                                    Function=BROWSE

.- Command Details ------------------------------------------------------------.
|                                                                              |
|  Command Name  T                                                             |
|                                                                              |
|  Parameters .. SYSNAME=&ZRMSYSNAME VERSION=&ZRMVERSION CLASS=17              |
|                NAME=&NAME                                                    |
```

SYSCMD Macro

The SYSCMD macro issues a system command and interprets the results. It lets you specify message rules to detect the required response message.

Parameters: SYSCMD Macro

Command

A mandatory field that specifies the system command issued by the SYSCMD macro.

The command follows the syntax of the SYSCMD command.

Jobname

Lets you limit the response messages to those from the specified job.

Wait Time

Specifies the maximum time, in seconds, that the region waits for a response message. The time is reset each time a response is received. The macro waits until a message that matches a rule in the Message Text fields is received or until the time expires without receiving any more responses.

Limits: A number from 1 through 9999.

Default: 30

Wait Time Expiry Return Code

Specifies the value of the macro return code if the expected message does not arrive before the expiry of the time specified in the Wait Time field.

Limits: A number from 1 through 999.

Default: 69

Message Text

Specifies the initial part of the required response message.

Limits: Up to 45 characters.

Return Code

Specifies the value set by the macro when it receives the response message.

Limits: A number from 0 through 999.

Extended Filter?

Indicates whether an extended filter has been defined. This field is empty until you complete the Message Text and Return Code fields. If you want to define an extended filter, enter S to access the Extended Message Filtering panel.

After this panel is completed, the Extended Filter field is updated automatically.

Returned Variables: SYSCMD Macro

&$RMMCMSGTEXT

Contains the full text of the message that satisfies the message rules.

&$RMMCWORD*n*

Contains the words in the message, where *n* indicates the position of the word in the message.

&ZRMMSG*n*

Contain any messages received before the required response message, where *n* is a number indicating the order in which those messages are received.

&$RMMCWTORID

Contains the ID of the last encountered WTOR message if the macro encounters any WTOR messages before it encounters an expected message.

For example, if the response to a command consists of three messages A, B, and C, and message B satisfies the message rules, then:

- &ZRMMSG1 contains message A
- &$RMMCMSGTEXT contains message B
- The macro ends on receipt of message B, and does not see message C

Example: SYSCMD Macro

In the following example, the SYSCMD macro issues the S DENMIMS1 command and awaits a specified response.

No extended filter has been defined in this example.

```
PROD----- Automation Services : SYSCMD Macro Parameter Definition -------------
Command ===>                                                  Function=BROWSE

.- System Command ------------------------------------------------------------.
|                                                                              |
| Command ..... S DENMIMS1                                                     |
|                                                                              |
| Jobname .....                                                                |
|                                                                              |
| Wait Time ... 5     Wait Time Expiry Return Code ... 0                       |
'------------------------------------------------------------------------------'
.- Response Message Analysis -------------------------------------------------.
|                                                        S=Extended Filter     |
|           Message Text                                   Return Extended     |
|                                                          Code   Filter?      |
|   ___    IEF403I DENMIMS1 DENMIMS1                         0      NO        |
|   ___    IEF450I DENMIMS1 DENMIMS1                         6      NO        |
|   ___                                                                        |
|   ___                                                                        |
|   ___                                                                        |
|                                                                              |
'------------------------------------------------------------------------------'
 F1=Help      F2=Split     F3=Exit
                           F9=Swap
```

TRANS Macro

The TRANS macro changes all occurrences of a set of characters in a text string into another set of characters.

Parameters: TRANS Macro

From

Specifies the list of characters you want to translate (replace). You must specify at least one character in this list.

To

Specifies the list of corresponding replacement characters.

String to be Translated-Input

Specifies the string that contains the characters you want translated.

The field is mandatory, and the entry may contain one or more variables. Translation occurs after the variables are substituted by their values.

Variable to hold Translated String-Output

Specifies the name of the variable to hold the translated string.

This is a mandatory field.

Example: TRANS Macro

In the following example, the TRANS macro examines the Input string:

```
AAA BBB CCC DDD EEE FFF GGG HHH
```

All occurrences of the specified characters are translated so that the result is:

```
111 222 333 444 555 FFF GGG HHH
```

This string is held in the variable &FRED1.

```
PROD------ Automation Services : TRANS Macro Parameter Definition -------------
Command ===>                                                    Function=BROWSE

.- What to Translate ------------------------------------------------------.
|                                                                           |
| From ..... A    To ....... 1   (All occurrences of the From character(s)  |
|      ..... B       ....... 2   in the Input string will be translated to  |
|      ..... C       ....... 3   the corresponding To character(s), and the |
|      ..... D       ....... 4   result will be stored in the Output string)|
|      ..... E       ....... 5                                              |
'---------------------------------------------------------------------------'

.- String to be Translated ------------------------------------------------.
|                                                                           |
| Input .... AAA BBB CCC DDD EEE FFF GGG HHH                                |
|           (Text String or Variable(s))                                    |
'---------------------------------------------------------------------------'

.- Variable to hold Translated String -------------------------------------.
|                                                                           |
| Output ... FRED1        (Variable Name)                                   |
'---------------------------------------------------------------------------'
 F1=Help      F2=Split     F3=Exit
                           F9=Swap
```

WAIT Macro

The WAIT macro instructs a process to wait for a specified period before executing the next step in the process.

Parameters: WAIT Macro

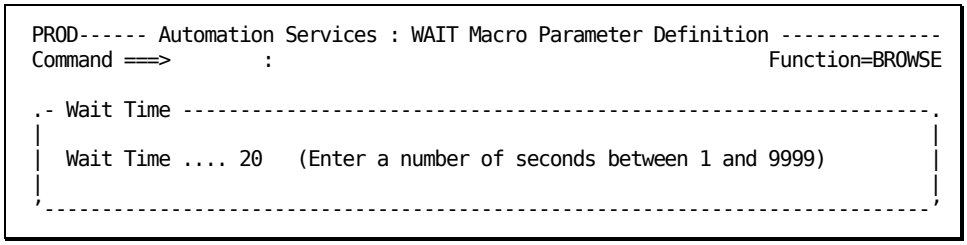
Wait Time

A mandatory field that lets you specify the length of time (in seconds) that the process is to wait before proceeding.

Limits: A number from 1 through 9999.

Example: WAIT Macro

In the following example, the WAIT macro instructs the process to wait 20 seconds before proceeding.



WAITEVNT Macro

The WAITEVNT macro waits for an N00102 event message.

Parameters: WAITEVNT Macro

Note: You can use the asterisk (*) as wild cards in the Name, Object, Resource, and Reference fields. A leading or embedded * represents a single character; a trailing * represents zero or more characters.

Name

Lets you use the name of the event as a criterion to select the event to wait for.

Type

Lets you use the type of event as a criterion to select the event to wait for. Valid values are as follows:

ALL

Selects events irrespective of their contents.

APPLICATION

Selects events that contain user-defined information.

ACCESS

Selects events that contain security alarm information.

CONFIGURATION

Selects events that contain object definition and relationship information.

PROCEDURAL

Selects events that contain scheduling and process control information.

SERVICEABILITY

Selects events that contain availability, degradation, error, fault, and recovery information.

UTILIZATION

Select events that contain accounting, performance, response time, and statistics information.

Scope

Lets you use the scope of delivery of the event as a criterion to select the event to wait for. Valid values are as follows:

REGION

Selects events that are delivered to processes within the region of the event issuer only.

SYSTEM

Selects events that are delivered to processes within the entire region.

Object

Lets you use the object class as a criterion to select the event to wait for.

Resource

Lets you use the name of the resource or resource pair (separated by a comma) for which the event is issued as a criterion to select the event to wait for.

Reference

Lets you use the reference code for the event as a criterion to select the event to wait for.

Wait Time

Specifies how long the macro can wait for an event that satisfies the specified criteria.

Return Codes: WAITEVNT Macro

0

Event message received.

4

Event message not received within the specified time.

Returned Variables: WAITEVNT Macro

&$RMMCEVENT1 through &$RMMCEVENT12

Contains the N00102 event message. The number of variables that contain value depends on the length of the message. Each variable can hold up to 256 bytes.

&$RMMCEVNAME

Contains the name of the event.

&$RMMCEVTYPE

Contains the type of the information in the event.

&$RMMCEVSCOPE

Contains the scope of delivery of the event.

&$RMMCEVOBJ

Contains the object class for the resource or resource pair for which the event is issued.

&$RMMCEVRES

Contains the name of the resource or resource pair for which the event is issued.

&$RMMCEVREF

Contains the reference code for the event.

&$RMMCEVDATA

Contains the part of data that is in the &$RMMCEVENT1 variable. If the data overflows into the other &$RMMCEVENT*n* variables, you can obtain all the data by concatenating the value in those variables to the value in this variable.

Example: WAITEVNT Macro

In the following example, the WAITEVNT macro waits for an event with a name that starts with OPERATIONS, and of the specified type and scope. The wait time period is 100 seconds.

```
PROD----- Automation Services : WAITEVNT Macro Parameter Definition -----------
Command ===>                                                  Function=BROWSE

.- Event Parameters ------------------------------------------------------------.
|                                                                               |
| Name......... OPERATIONS*                                                    |
|                                                                               |
| Type........+ SERVICEABILITY                                                 |
| Scope.......+ SYSTEM                                                         |
|                                                                               |
| Object.......                                                                |
| (Blanks will be translated to underscores)                                    |
|                                                                               |
| Resource.....                                                                |
| Reference....                                                                |
|                                                                               |
| Wait Time ... 100 (0 - 9999, Blank = Wait Forever)                           |
```

WAITSTAT Macro

The WAITSTAT macro waits for a specific resource or service state change and retrieves the status.

Parameters: WAITSTAT Macro

System Name/Version

Identifies the system image that owns the resource. If you do not specify an image, then during execution, the macro uses the following values:

If the macro is associated with a resource definition, the macro uses the image that owns the definition.

If the macro is not associated with a resource definition (for example, in an EventView rule), the macro uses the active local image.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Resource Class

Specifies the class name of the resource or service for which you want to detect the state change. This field can be completed manually, by entering the appropriate valid value, or automatically by selecting from the prompted field value list. To display the list, type **?** in the Resource Class field.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Resource Name

Specifies the name of the resource or service for which you want to detect the state change.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

State Type

Specifies whether the specified state change applies to the actual state, desired state, or the logical state. This field can be completed manually, by entering the appropriate valid type of state, or automatically by selecting from the prompted field value list. To display the list, type **?** in the State Type field.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Transition From State

Specifies the required state before the change. You can specify ANY to detect state changes from any state of the specified type. This field can be completed manually by entering the appropriate valid state, or automatically by selecting from the prompted field value list. To display the list, type **?** in the From State field.

If the current state does not satisfy this value, the macro cannot be processed and the return code is set to 8.

The values listed in the prompted field value list will vary depending on the value of the State Type field.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Transition To State

Specifies the required state after the change. This field can be completed manually by entering the appropriate valid state, or automatically by selecting from the prompted field value list. To display the list, type **?** in the To State field.

The values listed in the prompted field value list will vary depending on the value of the State Type field.

If the value of the From State field is ANY and the current state satisfies this value, the state change condition is deemed to be satisfied and the status is retrieved.

Note: You can specify a variable name in this field (for example, &VALUE) that resolves if passed as a parameter to this macro when executed.

Timeout Period

Specifies the maximum time in seconds that can elapse while waiting for the state change. After the specified time has expired, the return code is set to the value specified in the Return Code field.

Return Code

Specifies the return code to set if timeout occurs.

Limits: A number from 0 through 99. Do not specify 8 as the return code.

Variable Prefix

Specifies the prefix of the variables that contain the retrieved status information. The prefix lets you store the status information retrieved at different times in different variables.

The information is retrieved from the &ZRMST prefixed status variables when the specified state change is detected. The retrieved information is stored in a corresponding set of variables, with the ZRMST prefix replaced by the specified prefix, prefix. For example, the information retrieved from the &ZRMSTACTSTAT variable is stored in the &prefixACTSTAT variable.

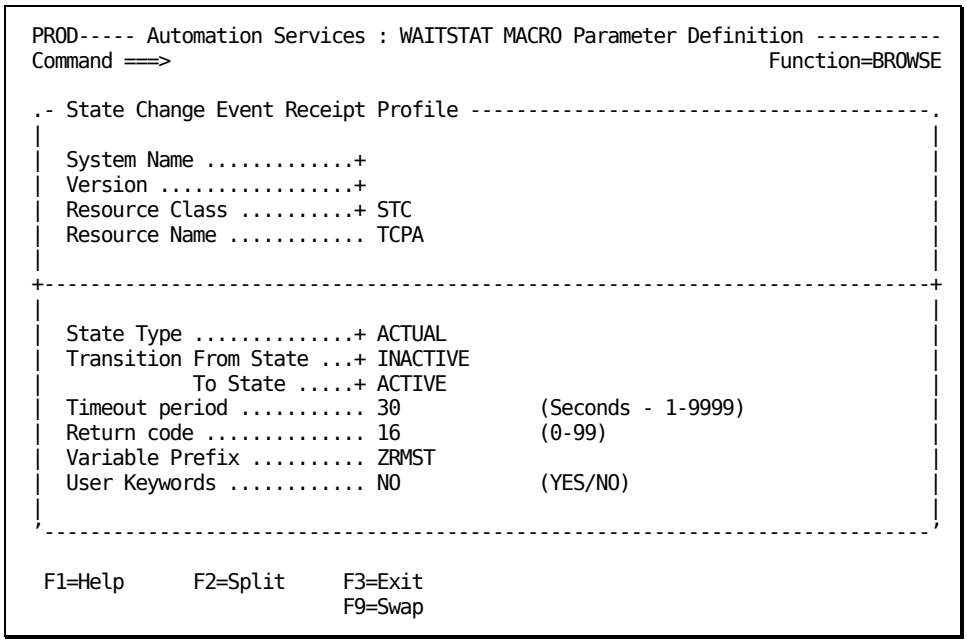
The prefix must not start with one of the following characters: # or $.

User Keywords

Lets you store the keyword that is specified in the User Defined Keyword field on the Define Extended Display Attribute panel of a triggered message rule in a resource definition. The keyword is stored in the &prefixUSRKEYW variable.

Example: WAITSTAT Macro

In the following example, you want to detect a change in the actual state of the resource TCPA from INACTIVE to ACTIVE. If the change is not detected within 30 seconds, the process sets the return code to 16. A variable prefix of ZRMST has been specified.



WTO Macro

The WTO macro sends a WTO message to local system consoles.

WTO Text

Specifies the WTO message to send.

Descriptor Codes

Specifies the descriptor codes assigned to the message. The value can be a single code, or a list of codes and code ranges (for example, 1,8-12).

Routing Codes

Specifies the routing codes assigned to the message. The value can be a single code, or a list of codes and code ranges (for example, 1,8-12).

Example: WTO Macro

In the following example, you send a message to consoles profiled for routing codes 1, 2, and 11 to indicate that the region is now managing systems network operations.

```
PROD-------- Automation Services : WTO Macro Parameter Definition -------------
Command ===                                                    Function=BROWSE

.- WTO Details ------------------------------------------------------------.
|                                                                           |
| WTO Text... THE REGION IS NOW MANAGING SYSTEM OPERATIONS.                 |
|                                                                           |
|                                                                           |
| Descriptor Codes...                                                       |
|                                                                           |
| Routing Codes...... 1,2,11                                                |
```

WTOR Macro

The WTOR macro sends a WTOR message to all local system consoles and waits for the reply.

Parameters: WTOR Macro

WTOR Text

Specifies the WTOR message to send.

Reply Match Text

Specifies the replies to look for. You can specify up to five reply text versions (rules), each setting a specific value for the macro return code. Specify enough text to positively identify each version. You can use the asterisk (*) as the wildcard character in the reply text.

R/Code

Specifies the value to set for the return code if a reply satisfying the rule in the Reply Match Text field is received. Use different values to differentiate between the different replies. The macro return code thus indicates the type of reply received.

Store Actual Reply Text in Variable Name

Specifies the name of the variable in which to put the received reply.

Wait Time

Specifies the number of seconds to wait for a reply that satisfies one of the rules in the Reply Match Text field. If the specified time expires, the WTOR message is canceled and the return code is set to the value specified in the Wait Time Expiry Return Code field.

Wait Time Expiry Return Code

Specifies the value to set for the return code if no valid reply is received within the specified time.

Example: WTOR Macro

In the following example, you want to find out whether one of five specified users has logged on to a console.

```
PROD----- Automation Services : WTOR Issue MACRO Parameter Definition ---------
Command ===                                                    Function=BROWSE

.- WTOR Text To Be Issued ------------------------------------------------.
|                                                                          |
| WTOR Text... WHAT IS YOUR USER ID?                                       |
|                                                                          |
'--------------------------------------------------------------------------'
.- Response Text Analysis ------------------------------------------------.
|                                                                          |
|             Reply Match Text                              R/Code         |
|             USER01                                        10             |
|             USER02                                        20             |
|             USER03                                        30             |
|             USER04                                        40             |
|             USER05                                        50             |
|                                                                          |
|             Store Actual Reply Text in Variable Name  REPLYTEXT          |
|                                                                          |
|             Wait Time 120   Wait Time Expiry Return Code 32              |
'--------------------------------------------------------------------------'

 F1=Help      F2=Split     F3=Exit
                           F9=Swap
```

Appendix K: Data Set Descriptions

This section contains the following topics:

Data Set Types (see page 467)
Installation (see page 469)
Product Components (see page 470)
Management Services Data Sets (see page 470)
PDSE Services Data Sets (see page 476)
Automation Services Data Sets (see page 477)
SOLVE Subsystem Interface Data Sets (see page 477)
TCP/IP Services Data Sets (see page 478)
File Transfer Services Data Sets (see page 479)
SNA Services Data Sets (see page 482)
SNA Automation Services Data Sets (see page 484)
FTS Data Sets (see page 485)
ReportCenter Data Sets (see page 487)
WebCenter SDK Data Sets (see page 487)

Data Set Types

In the following tables, the Types column contains information about the types of data sets.

The following table explains the data set types:

Types	Explanation
Run-time PDS	Shared non-VSAM SMP target files that are allocated during installation, and used by an active region, for example, *dsnpref*.NMC0.*ccdsname*.
Run-time PDSE	Shared non-VSAM SMP target files that are allocated during installation and used by an active region, for example, *dsnpref*.NMC0.*ccdsname*. Program objects that must be executed from a PDSE are stored in these files.
Other PDS	Shared non-VSAM PDS created separately to this product.
Run-time PDS (external)	Shared non-VSAM SMP target libraries that are not used by the region.

Types	Explanation
Shared run-time VSAM (MODSDIS / NETINFO / NSCNTL / PANLDIS / OSCNTL / UAMS)	Files that are identified by the unique data set prefixes you enter when you set up your regions. You can choose to select a data set prefix for each file. You can make the data set prefix the same, or different from those you have used for other files. These run-time files are used by an active region. These files will be shared by more than one region. VSAM data sets that can be shared in this manner are defined using SHAREOPTIONS(3,3).
Distribution	SMP distribution libraries that are used during the installation and maintenance of a product, for example, *dsnpref*.NMC0.*ccdsname*.
Local Sequential	Run-time sequential files that are allocated during setup to be used by an active region, and cannot be shared between multiple regions, for example, *dsnpref.rname.dsname.*
Staging	Shared non-VSAM files that are used during the installation process to store sequential copies of files including the product panels, MODS, and OSCNTL files, for example, *dsnpref*.NMC0.*ccdsname.*S. These sequential copies are then merged to create the run-time files. If you intend to create additional regions at a later date, you can use the staging files as input to the setup process.
Local PDS	Run-time PDS files that are allocated during setup to be used by an active region, and cannot be shared between multiple regions, for example, *dsnpref.rname.dsname*.
Local VSAM	Run-time VSAM files that are allocated during setup to be used by an active region, and cannot be shared between multiple regions, for example, *dsnpref.rname.dsname*.
Shared PDS	Run-time PDS files that are allocated during setup to be used by an active region, and can be shared between multiple regions, for example, *dsnpref.*NMC0*.dsname*.
TESTEXEC	File that is identified by the unique data set prefix you enter when you set up your region.

Installation

Data Sets	Purpose	Type	Run-time DDNames
*dsnpref.*NMC0.INSTDB	The INSTDB data set stores your site-specific installation values, which can be reused in future installations of products in the same suite. The installation software allocates this data set the first time you perform an installation. This data set is also used by the setup and maintenance software.	-	-
*dsnpref.*NMC0.INSTALL.JCL	After the installation software has collected your site-specific installation values, it generates the installation JCL. Before generating the JCL, you must specify the library where you want to store the generated JCL. INSTALL.JCL is the default JCL library. You must use an empty data set each time you perform the installation. This ensures that the jobs in your JCL library are the only ones required for the current installation.	-	-
*dsnpref.*NMC0.*rname*.JCL *dsnpref.*NMC0.*ssiname*.JCL	After the setup software has collected the setup values for your region or SOLVE SSI, it generates the setup JCL. These data sets contain the setup jobs, including the members that contain the execution JCL required to run the product components.	-	-
dsnpref.NMC0.VTAM.JCL	After the Create VTAM Definitions software has collected your values, it generates the JCL. This data set contains the generated VTAM major node and JCL to assemble mode tables.	-	-
*dsnpref.*NMC0.FIX.SP*n*.JCL *dsnpref.*NMC0.FIX.DASD.JCL *dsnpref*.NMC0.FIX.WEBCENTR.JCL	After the maintenance software has collected your maintenance values, it generates maintenance JCL. This data set contains the maintenance jobs that you run to apply maintenance to your installed products.	-	-

Product Components

Your product operates with a combination of common and product-specific components.

The installation process and these components have data set requirements.

Management Services Data Sets

AC2DEXEC

Is the SMP DLIB that contains the same distributed NCL procedures as the CC2DEXEC run-time library.

Type: Distribution

AC2DJCL

Is the SMP DLIB that contains the same members as the CAIJCL installation data set.

Type: Distribution

AC2DMAC

Is the SMP DLIB that contains the same information as the CC2DMAC library.

Type: Distribution

AC2DMOD

Is the SMP DLIB that contains the following modules:

- Modules used by SMP to build the load modules in the run-time LOAD libraries and the TSO CMDLIB command library (for the TSO interface).
- The distributed CA Datacom program modules for ReportCenter, for example, the ADATACOM run-time libraries.
- The TCP/IP Services modules. SMP uses the modules in this library to build the load modules in the run-time load library.
- The File Transfer Services modules.

Type: Distribution

AC2DSAMP

Is the SMP DLIB that contains the same information as the CC2DSAMP library.

Type: Distribution

AC2DVSMI

Is the SMP DLIB that contains the same information as the CC2DVSMI library.

Type: Distribution

ALERTH

Is part of the Alert Monitor and enables you to view alerts that have been created in the past. Alerts in the history file can contain notes showing manual and automated actions that were performed for the alert.

Type: Local VSAM

Run-time DDname: ALERTH

CAIJCL

Is the PDS that contains the installation and maintenance JCL members.

Type: Run-time PDS

CAILINK

Is a PDS run-time load library that contains various NetView exits. Include in the STEPLIB DD concatenation for NetView if you are using the NVC component.

If your auditors want CA Auditor Product Descriptor Module (PDM) to know of your product running on your system, include this library in the system linklist.

Type: Run-time PDS

CAILPA

Is a separate PDS run-time load library containing modules that must be executed from the LPA. The data set contains only CNMNETM plus its aliases.

Type: Run-time PDS

CC2DEXEC

Is the PDS that contains distributed NCL procedures and is concatenated after the TESTEXEC data set in DD COMMANDS. Collectively these data sets make up the procedure library.

Members in CC2DEXEC must not be changed.

Important! If modifications are required, we recommend that you create an SMP/E ++USERMOD to record and control the changes, and then copy the member to TESTEXEC. Alternatively, you can copy the distributed member to the region's TESTEXEC data set for modification.

Type: Run-time PDS (external)

Run-time DDName: COMMANDS

CC2DLMD0

Is the PDS into which the TSO interface load modules are linked during installation.

Type: Run-time PDS (external)

Run-time DDName: STEPLIB (TSO)

CC2DLOAD

Is the load library. This load library must be APF-authorized. This means that the run-time load library, as referenced by the STEPLIB DD statement, must be defined in the operating system APF list. If no STEPLIB DD statement is used, the program load modules must reside in one of the existing authorized linklist libraries. Ensure that these requirements are met before attempting to start your region.

You can install your product into its own installation load library and copy the load modules across to the system load library.

The library also contains the CA Datacom load modules for ReportCenter.

Type: Run-time PDS

Run-time DDName: STEPLIB

CC2DMAC

Is the PDS that contains macros and copybooks for the sample assembler programs, which are also distributed in source form in the AC2DMAC library.

Type: Run-time PDS (external)

CC2DSAMP

Is the PDS that contains various sample exits, utilities, source code, and JCL. The members contain documentation.

Type: Run-time PDS (external)

CC2DVSMI

Is the interim staging data set to which MODS, OSCNTL, PANEL, NETINFO, ICOPANL, and RAMDB files are unloaded. They are later copied to the run-time MODSDIS, OSCNTL, PANLDIS, NETINFO, ICOPANL, and RAMDB files.

Type: Staging

FMTDUMP

Is the data set the region uses for a formatted dump.

Type: SYSOUT

Run-time DDName: FMTDUMP

LOG

Is the data set the region uses for the hardcopy log.

Type: SYSOUT

Run-time DDNames: LOG1 through LOG9

MODSDIS

Is the VSAM data set that contains the MODS database. The region uses the MODS database for control information for processing that includes standard message information, online help text, menus, Print Services Manager (PSM) definitions, and Report Writer definitions.

Type: Shared run-time VSAM

Run-time DDName: MODSDIS

Note: For information about the allocation and usage of MODS databases, see the *Managed Object Development Services Programmer and Administrator Guide*.

MODSUSR

Is a data set that has the same functions as MODSDIS and is intended for your own records. You can have separate records for test and production environments.

Type: Local VSAM

Run-time DDName: MODSUSR

NETINFO

Is the VSAM data set that contains various error codes (for example, 3274 error codes and SNA sense codes).

Type: Shared run-time VSAM

Run-time DDName: NETINFO

NML0G01 through NMLOG03

Are the VSAM data sets to which the region's LOGPROC NCL procedure ($LOPROC) writes all activity log messages for subsequent online browsing from a terminal.

Type: Local VSAM

Run-time DDNames: NMLOG01 through NMLOG03

OSCNTL

Is the VSAM data set used by the region to store MDO definitions and compiled object class specifications.

Type: Shared run-time VSAM

Run-time DDName: OSCNTL

PANLDIS

Is the VSAM data set that stores full-screen panels defined by using the online editor. These panels are used by NCL procedures and Panel Services.

Your product provides the facility for multiple panel data sets per region, allowing the data sets to be concatenated and different data sets to be available to different users.

Type: Shared run-time VSAM

Run-time DDName: PANLDIS

PANLUSR

Is a data set that has the same functions as PANLDIS and is intended for your own records. You can have separate records for test and production environments.

Type: Local VSAM

Run-time DDName: PANLUSR

PARMLIB

Is the data set (*dsnpref*.NMC0.PARMLIB) that contains the setup parameters for the product components in a region.

Type: Shared PDS

Run-time DDNames: DSIPARM and SXCTL

PSPOOL

Is the data set used to store printed output handled by the PSM facility.

Type: Local VSAM

Run-time DDName: PSPOOL

REXXAN

Is the data set used to store analysis output from the REXX Analyzer.

Type: Local PDS

REXXREP

Is the data set used to store the report generated by the REXX Analyzer.

Type: Local sequential

TESTEXEC

Is the data set (*dsnpref.rname*.TESTEXEC) that is concatenated as the first of the data sets forming the COMMANDS DD. The data set is used for the following procedures:

- User-written NCL procedures
- Modified versions of supplied procedures that have been copied from the distributed libraries
- Installation modified INIT and READY procedures

The setup process allocates TESTEXEC as a local data set.

Type: TESTEXEC

Run-time DDName: COMMANDS

UAMS

Is the User ID Access Maintenance Subsystem (UAMS) data set containing the security definitions for users authorized to use the region.

In a shared DASD environment where multiple regions are operative, perhaps connected by Inter-Network Management Connection (INMC), you can define one UAMS data set that all regions can share. The regions use reserve and release logic during accesses, and ensure the integrity of the data set in a shared DASD environment. If only one data set is used, operators using the Remote Operator Facility (ROF) to connect to another region have identical authority and privileges in both regions. This situation may not be satisfactory if one region is dedicated to testing and the other to production. In this case, use two UAMS data sets, allowing a user to be profiled differently in the two regions.

If the installation uses a security exit to replace the UAMS component entirely, the UAMS data set is not required.

When logging on or changing UAMS records, a RESERVE is issued for NMUAMS. The RESERVE has the following attributes:

- Qname: NMUAMS (padded to eight characters with blanks)
- Rname: 44-character blank padded UAMS data set name
- Scope: SYSTEMS
- Level: EXCLUSIVE
- UCB: UAMS UCM address

If the sysplex is in a STAR configuration, you can convert this RESERVE to a GRS global ENQ. If the sysplex is a RING configuration, do not convert the RESERVE.

Type: Shared run-time VSAM

Run-time DDName: USERIDS

VFS

Is the data set used by the region for internal processing activities and as a general database for use by the various products. The VFS is a common database used to store many record types and must not be shared across regions.

Type: Local VSAM

Run-time DDName: VFS

PDSE Services Data Sets

AC2DMODE

Is the SMP DLIB that contains modules used by SMP to build the program objects in the run-time PDSE.

Type: Distribution

CC2DPLD

Is a partitioned data set extended run-time library that must be APF-authorized. The data set contains program objects.

Type: Run-time PDSE

Run-time DDName: STEPLIB

Automation Services Data Sets

ICOPANL

Is the data set that stores Automation Services icon panels.

Type: Local VSAM

Run-time DDname: ICOPANL

RAMDB

Is the distributed database that stores definitions for Automation Services.

Type: Local VSAM

Run-time DDname: RAMDB

RAMDBST

Is the data set used by Automation Services in a multisystem environment to house database updates before they are propagated to linked regions.

Type: Local VSAM

Run-time DDname: RAMDBST

RAMDBWK

Is the work data set used by Automation Services to assist the link and synchronization process.

Type: Local VSAM

Run-time DDname: RAMWORK

SOLVE Subsystem Interface Data Sets

SSIDB

Is the data set that acts as a disk-based backup to the nonvolatile part of the SOLVE SSI database. The data set contains saved application data and allows data to be preserved across restarts.

Type: Local VSAM

Run-time DDname: SSIDB

SSIPARM

Is the data set that contains the setup members for the SOLVE SSI.

Type: Run-time PDS

Run-time DDname: SSIIN

TCP/IP Services Data Sets

AC11EXEC

Is the SMP DLIB that contains the same distributed NCL procedures as in the CC11EXEC run-time libraries.

Type: Distribution

AC11VSMI

Is the SMP DLIB that contains the same information as the CC11VSMI library.

Type: Distribution

AC11XML

Is the SMP DLIB that contains the same Extensible Markup Language (XML) data as the CC11XML library.

Type: Distribution

CC11EXEC

Is the PDS that contains distributed NCL procedures and is concatenated after the TESTEXEC data set in DD COMMANDS. Collectively these data sets make up the procedure library.

Members must not be changed.

Important! If modifications are required, we recommend that you create an SMP/E ++USERMOD to record and control the changes, and then copy the member to TESTEXEC. Alternatively, you can copy the distributed member to the region's TESTEXEC data set for modification.

Type: Run-time PDS

Run-time DDName: COMMANDS

CC11VSMI

Is the interim staging data set to which MODS, OSCNTL, and PANEL files are unloaded. They are later copied to the run-time MODSDIS, OSCNTL, and PANLDIS files.

Type: Staging

CC11XML

Is the PDS that contains the XML data used by CA MSM.

Type: Run-time PDS (external)

IPFILE

Is the data set used for storing trend data, packet trace data, and other information. The product implementation process allocates the data set.

Type: Local VSAM

Run-time DDName: IPFILE

IPLOG

Is the data set used to store reported events.

Type: Local VSAM

Run-time DDName: IPLOG

IPLOGSEQ

Is the data set used in the IPLOG reorganization process. The process copies all records from the IPLOG NDB out to the IPLOGSEQ data set and then copies them all back. The IPLOG NDB is reset in the process.

Type: Local sequential

Run-time DDName: IPLOGSEQ

IPMIBX

Is the management information base (MIB) import/export data set used by MIBinsight.

OBEYFILE

Is a sequential work file used to store OBEYFILE commands submitted to Communications Server (for example, packet trace).

Type: Local PDS

SYSPRINT

Is a sequential work file used by the NETSTAT and OBEYDSN commands. The CA NetMaster NM for TCP/IP implementation process allocates the data set to the region.

Type: Local sequential

File Transfer Services Data Sets

AC17EXEC

Is the SMP DLIB that contains distributed NCL procedures as in the CC17EXEC run-time library.

Type: Distribution

AC17MAC

Is the SMP DLIB that contains the same information as the CC17MAC library.

Type: Distribution

AC17SAMP

Is the SMP DLIB that contains the same information as the CC17SAMP library.

Type: Distribution

AC17VSMI

Is the SMP DLIB that contains the same information as the CC17VSMI library.

Type: Distribution

AC17XML

Is the SMP DLIB that contains the same XML data as the CC17XML library.

Type: Distribution

CC17EXEC

Is the PDS that contains distributed NCL procedures and is concatenated after the TESTEXEC data set in DD COMMANDS. Collectively these data sets make up the procedure library.

These members must not be changed.

Important! If modifications are required, we recommend that you create an SMP/E ++USERMOD to record and control the changes, and then copy the member to TESTEXEC. Alternatively, you can copy the distributed member to the region's TESTEXEC data set for modification.

Type: Run-time PDS

Run-time DDName: COMMANDS

CC17MAC

Is the PDS that contains macros and copybooks required for the sample assembler programs, which are also distributed in source form in the CC17SAMP library.

Type: Run-time PDS (external)

CC17SAMP

Is the PDS that contains various sample exits, utilities, source code, and JCL. The members contain documentation.

Type: Run-time PDS (external)

CC17VSMI

Is the interim staging data set to which MODS, OSCNTL, and PANEL files are unloaded. They are later copied to the run-time MODSDIS, OSCNTL, and PANLDIS files.

Type: Staging

CC17XML

Is the PDS that contains the XML data used by CA MSM.

Type: Run-time PDS (external)

EVNTARC

Is the data set used to archive reported events stored in the EVNTDB data set. This data set is allocated during setup time but is not populated unless archiving is specified during product implementation.

Type: Local sequential

Run-time DDName: EVNTARC

EVNTDB

Is the data set used to store reported events.

Type: Local VSAM

Run-time DDName: EVNTDB

EVNTSEQ

Is the data set used to reorganize the EVNTDB data set.

Type: Local sequential

Run-time DDName: EVNTSEQ

FTLOG0*n*

Are log files for browsing file transfer activity at a terminal.

Type: Local VSAM

Run-time DDNames: FTLOG0*n*

FTSCHD

Is the data set that stores file transfer schedule events so that if the region restarts before a schedule completes successfully, the events for that schedule can still be displayed.

Type: Local VSAM

Run-time DDName: FTSCHD

SNA Services Data Sets

AC2AEXEC

Is the SMP DLIB that contains the distributed NCL procedures as in the CC2AEXEC run-time libraries.

Type: Distribution

AC2AMAC

Is the SMP DLIB that contains the same information as the CC2AMAC library.

Type: Distribution

AC2ASAMP

Is the SMP DLIB that contains various sample exits, utilities, source code, and JCL. The members contain documentation.

Type: Distribution

AC2AVSMI

Is the SMP DLIB that contains the same information as the CC2AVSMI library.

Type: Distribution

AC2AXML

Is the SMP DLIB that contains the same XML data as the CC2AXML library.

Type: Distribution

CC2AEXEC

Is the PDS that contains distributed NCL procedures and is concatenated after the TESTEXEC data set in DD COMMANDS. Collectively these data sets make up the procedure library.

Members must not be changed.

Important! If modifications are required, we recommend that you create an SMP/E ++USERMOD to record and control the changes, and then copy the member to TESTEXEC. Alternatively, you can copy the distributed member to the region's TESTEXEC data set for modification.

Type: Run-time PDS

Run-time DDName: COMMANDS

CC2AMAC

Is the PDS that contains macros and copybooks for the sample assembler programs.

Type: Runtime PDS (external)

CC2ASAMP

Is the PDS that contains various sample exits, utilities, source code, and JCL. The members contain documentation.

Type: Run-time PDS (external)

CC2AVSMI

Is the interim staging data set to which NSCNTL, MODS, OSCNTL, and PANEL files are unloaded. They are later copied to the run-time NSCNTL, MODSDIS, OSCNTL, and PANLDIS files.

Type: Staging

CC2AXML

Is the PDS that contains the XML data used by CA MSM.

Type: Run-time PDS (external)

NEWSBKP

Is an ESDS or sequential data set used during online reorganization of the Network Error Warning System (NEWS) database. The data set must be large enough to contain an unloaded copy of the NEWS database.

Note: This data set is required only when the NEWS database has to be reorganized by using the online reorganization function. If you are short on disk space, you can dynamically allocate it when needed.

Type: Local VSAM

Run-time DDName: NEWSBKP

NEWSFIL

Is a data set used by NEWS to log records received across the CNM interface from VTAM for review and analysis. To cater for the condition when this data set fills up, we recommend that you allocate this database with REUSE to enable reorganization.

The $NSINIT procedure, which is invoked from NMINIT, dynamically allocates the DD statement for NEWSFIL and the associated data set. You can customize this procedure to suit individual installation standards or allocate the data set in the region started task.

Type: Local VSAM

Run-time DDName: NEWSFIL

NSCNTL

Is the Network Management Control file used by NEWS to process CNM request/response units (RUs) at the arrival and display processing stages.

Type: Shared run-time VSAM

Run-time DDName: NSCNTL

NTSLOG

Is the data set for logging session details processed by the Network Tracking System (NTS). The size of this data set is dependent on the size of the network, and the amount and type of the logged data.

Type: Local VSAM

Run-time DDName: NTSLOG

SNA Automation Services Data Sets

AC18EXEC

Is the SMP DLIB that contains distributed NCL procedures as in the CC18EXEC run-time library.

Type: Distribution

AC18VSMI

Is the SMP DLIB that contains the same information as CC18VSMI.

Type: Distribution

AC18XML

Is the SMP DLIB that contains the same XML data as the CC18XML library.

Type: Distribution

CC18EXEC

Is the PDS that contains distributed NCL procedures and is concatenated after the TESTEXEC data set in DD COMMANDS. Collectively these data sets make up the procedure library.

Members must not be changed.

Important! If modifications are required, we recommend that you create an SMP/E ++USERMOD to record and control the changes, and then copy the member to TESTEXEC. Alternatively, you can copy the distributed member to the region's TESTEXEC data set for modification.

Type: Run-time PDS

Run-time DDName: COMMANDS

CC18VSMI

Is the interim staging data set to which MODS, OSCNTL, and PANEL files are unloaded. They are later copied to the run-time MODSDIS, OSCNTL, and PANLDIS files.

Type: Staging

CC18XML

Is the PDS that contains the XML data used by CA MSM.

Type: Run-time PDS (external)

RSBIWKF

Is the network discovery work file.

RSDB

Is a database used by the CA NetMaster NA product.

Type: Local VSAM

Run-time DDName: RSDB

FTS Data Sets

ADEMCLS0

Is the SMP DLIB that contains the same members as the CDEMCLS0 library.

Type: Distribution

ADEMEXEC

Is the SMP DLIB that contains the distributed NCL procedures as in the CDEMEXEC run-time libraries.

Type: Distribution

ADEMMSG0

Is the SMP DLIB that contains the same members as the CDEMMSG0 library.

Type: Distribution

ADEMPNL0

Is the SMP DLIB that contains the same members as the CDEMPNL0 library.

Type: Distribution

ADEMVSMI

Is the SMP DLIB that contains the same information as the CDEMVSMI library.

Type: Distribution

ADEMXML

Is the SMP DLIB that contains the same XML data as the CDEMXML library.

Type: Distribution

CDEMCLS0

Is the PDS that contains CLISTs.

Type: Run-time PDS (external)

CDEMEXEC

Is the PDS that contains distributed NCL procedures and is concatenated after the TESTEXEC data set in DD COMMANDS. Collectively these data sets make up the procedure library.

Members must not be changed.

Important! If modifications are required, we recommend that you create an SMP/E ++USERMOD to record and control the changes, and then copy the member to TESTEXEC. Alternatively, you can copy the distributed member to the region's TESTEXEC data set for modification.

Type: Run-time PDS

Run-time DDName: COMMANDS

CDEMMSG0

Is the PDS that contains ISPF messages.

Type: Run-time PDS (external)

CDEMPNL0

Is the PDS that contains ISPF panels.

Type: Run-time PDS (external)

CDEMVSMI

Is the interim staging data set to which MODS, OSCNTL, and PANEL files are unloaded. They are later copied to the run-time MODSDIS, OSCNTL, and PANLDIS files.

Type: Staging

CDEMXML

Is the PDS that contains the XML data used by CA MSM.

Type: Run-time PDS (external)

ReportCenter Data Sets

AC2DHFSR

Is the PDS that contains z/OS UNIX file system file elements for ReportCenter. Each element is stored as a separate PDS member.

Do not change these distributed files.

Type: Distribution

CC2DHFSR

Is the DDDEF that defines the *iia-prefix/nm*/reporter/ z/OS UNIX file system directory path name that contains distributed files such as code, logos, style sheet, and report definitions.

Type: Run-time z/OS UNIX files

WebCenter SDK Data Sets

AC2DHFSS

Is the PDS that contains z/OS UNIX file system file elements for the WebCenter software development kit (SDK). Each element is stored as a separate PDS member.

Do not change these distributed files.

Type: Distribution

CC2DHFSS

Is the DDDEF that defines the *iia-prefix/nm*/sdk/ z/OS UNIX file system directory path name that contains distributed files such as code, logos, style sheet, and report definitions.

Type: Run-time z/OS UNIX files

Appendix L: Distributed Knowledge Base Changes

If you have previously customized any of the changed objects or are using any of the deleted objects, review each instance in your old knowledge base and make modifications in your new knowledge base as required.

Note: For more information about knowledge base migration, see the *Installation Guide*.

This section contains the following topics:

$TEMPLAT 0001 Definitions (see page 489)
$TEMPLAT 0002 Definitions (see page 492)
$TEMPLAT 0003 Definitions (see page 495)
Commands (see page 496)
File Transfer Rules (see page 499)
Global Processes (see page 500)
IP Node Monitor Groups (see page 500)
Macros (see page 501)
Monitoring Attributes (see page 501)
Prompt Lists (see page 508)
Alert Monitor Filters (see page 509)
Status Monitor Filters (see page 509)
VTAM States (see page 510)
Resource Groups (see page 511)
Icon Panels (see page 511)

$TEMPLAT 0001 Definitions

$TEMPLAT 0001 definitions have changed in r12, r11.6 SP1, and r11.6.

r12 $TEMPLAT 0001 Definition Changes

The following list shows the new, changed, or deleted definitions for r12:

New

```
ASMON   :DB2              - DB2 (DDF) Address Space
```

Changed

```
ASMON   :STC              - Address Space Monitor
OSA     :DIRECTXPRESS     - OSA Direct (OSASF,SNMP)
OSA     :EXPRESS          - OSA Express (OSASF,SNMP)
OSA     :OSA2             - OSA-2 (OSASF)
STACK   :IBM              - TCP/IP Communications Server
STACK   :IBMV6            - TCP/IP CS enabled for IPv6
```

Deleted

```
ROUTER  :2216ROUTER       - IBM 2216 Router
```

r11.6 SP1 $TEMPLAT 0001 Definition Changes

The following list shows the new, changed, or deleted definitions for r11.6 SP1:

New

```
APPNHPR :APPNHPR          - APPN High Performance
```

Changed

```
EE      :EE               - Enterprise Extender
```

Deleted

No definitions have been deleted.

r11.6 $TEMPLAT 0001 Definition Changes

The following list shows the new, changed, or deleted definitions in r11.6:

New

```
APPNHPR :APPNHPR          - APPN High Performance Routing
ASMON   :TNSERVER         - IBM Telnet Server
STACK   :IBMV6            - TCP/IP CS enabled for IPv6
```

Changed

```
ASMON   :STC              - Address Space Monitor
CDMGR   :CDJOB            - CONNECT:Direct JOB
CDMGR   :CDNT             - NT CONNECT:Direct Manager
CDMGR   :CDOS400          - OS/400 CONNECT:Direct Manager
CDMGR   :CDSTC            - CONNECT:Direct Started Task
CDMGR   :CDTANDEM         - TANDEM CONNECT:Direct Manager
```

```
CDMGR   :CDUNIX           - UNIX CONNECT:Direct Manager
CDMON   :ACTIVEQ          - C:D Active Queue Monitor
CDMON   :BADQ             - C:D Bad Queue Monitor
CDMON   :CALLQ            - C:D Call Queue Monitor
CDMON   :CONNECTIONS      - C:D TCP/IP Connection Monitor
CDMON   :EXECQ            - C:D Exec Queue Monitor
CDMON   :HOLDQ            - C:D Hold Queue Monitor
CDMON   :HOLDQWC          - C:D Hold Queue with WC Status
CDMON   :INITIALQ         - C:D Initial Queue Monitor
CDMON   :LISTENER         - C:D TCP/IP Listener Task Mon.
CDMON   :REMOTENODE       - C:D Remote Node Monitor
CDMON   :RETAINQ          - C:D Retain Queue Monitor
CDMON   :STARTQ           - C:D Start Queue Monitor
CDMON   :SUSPENDQ         - C:D Suspend Queue Monitor
CDMON   :TIMERQ           - C:D Timer Queue Monitor
CDMON   :TIMERQRETRY      - C:D Timer Queue - Retry Status
CDMON   :TRANSFER         - C:D Transfer Monitor
CDMON   :WAITQ            - C:D Wait Queue Monitor
CDMON   :XRTQ             - C:D Retain-Timer Queue Monitor
CICMON  :CICS             - CICS Monitor
CIP     :CIP              - Cisco Channel Intf Processor
CMMGR   :CMSTC            - C:Mailbox Started Task
CMMON   :ACQUEUE          - Queued Auto Connect Monitor
CMMON   :BSCLINES         - BSC Line Monitor
CMMON   :SNASESS          - Stalled SNA Session Monitor
CSM     :CSM              - Communications Storage Manager
EE      :EE               - Enterprise Extender
FTPMGR  :AXSJOB           - FTP TCPAccess Server - JOB
FTPMGR  :AXSSTC           - FTP TCPAccess Server - STC
FTPMGR  :AXS52JOB         - FTP TCPAccess V5.2 Server  JOB
FTPMGR  :AXS52STC         - FTP TCPAccess V5.2 Server  STC
FTPMGR  :AXS53JOB         - FTP TCPAccess V5.3 Server  JOB
FTPMGR  :AXS53STC         - FTP TCPAccess V5.3 Server  STC
FTPMGR  :CSFTPSRV         - Comms. Server FTP Server - STC
FTPMGR  :SFTPJOB          - TCPaccess FTP Server - JOB
FTPMGR  :SFTPSTC          - TCPaccess FTP Server - STC
FTPMON  :CONNECTIONS      - FTP TCP/IP Connections Monitor
FTPMON  :IPLISTEN         - FTP Listener Port Monitor
FTPMON  :REMOTENODE       - FTP Remote Node Monitor
FTSMGR  :FTSJOB           - FTS JOB
FTSMGR  :FTSREMOTE        - FTS running on a remote system
FTSMGR  :FTSSELF          - FTS running in this region
FTSMGR  :FTSSTC           - FTS Started Task
FTSMON  :LINKMONITOR      - FTS Link Monitor
INTNL   :$RMCCISR         - Monitor ISR Manager
INTNL   :$RMEV00H         - Monitor Event Handler
INTNL   :$RMIM00H         - Monitor ICON Manager
INTNL   :$RMMF00H         - Monitor SSI Message Filter
INTNL   :$RMMF01S         - Monitor Secondary AOMPROC
```

```
INTNL   :APPCLINK          - APPC Link Definition
INTNL   :NMAGENT           - Unicenter Agent
NCPMON  :NCP               - NCP template
OSA     :DIRECTNOSF        - OSA Direct (No OSASF,No SNMP)
OSA     :DIRECTXPRESS      - OSA Direct (OSASF,SNMP)
OSA     :EXPRESS           - OSA Express (OSASF,SNMP)
OSA     :EXPRESSNOSF       - OSA Express (No OSASF,No SNMP)
OSA     :OSA2              - OSA-2 (OSASF)
OSA     :OSA2NOSF          - OSA-2 (No OSASF)
ROUTER  :2216ROUTER        - IBM 2216 Router
STACK   :IBM               - TCP/IP Communications Server
STACK   :TCPACCES          - TCPaccess
VIPA    :DEFAULT           - Standard dynamic VIPA
VIPA    :DEFAULTDIST       - Sysplex distributor VIPA
XCMGR   :XCJOB             - XCOM as a JOB
XCMGR   :XCSTC             - XCOM Started Task
XCMON   :ACTIVE            - Active Transfer Monitor
XCMON   :CONNECTIONS       - Connections Monitor
XCMON   :HELD              - Held Transfer Monitor
XCMON   :INACTIVE          - Inactive Transfer Monitor
XCMON   :LISTENER          - Listener Monitor
XCMON   :REMOTENODE        - Remote Node Monitor
XCMON   :STALLED           - Stalled Transfer Monitor
XCMON   :SUSPENDED         - Suspended Transfer Monitor
PRCESS  :RFCDHCK1          - CDMGR Health Check - Poll C:D Stats Exit
```

Deleted

No definitions have been deleted.

$TEMPLAT 0002 Definitions

$TEMPLAT 0002 definitions have changed in r12, r11.6 SP1, and r11.6.

r12 $TEMPLAT 0002 Definition Changes

The following list shows the new, changed, or deleted definitions for r12:

New

```
ASMON   :DB2              - DB2 (DDF) Address Space
```

Changed

```
ASMON   :STC              - Address Space Monitor
OSA     :DIRECTXPRESS     - OSA Direct (OSASF,SNMP)
OSA     :EXPRESS          - OSA Express (OSASF,SNMP)
OSA     :OSA2             - OSA-2 (OSASF)
STACK   :IBM              - TCP/IP Communications Server
STACK   :IBMV6            - TCP/IP CS enabled for IPv6
```

Deleted

```
ROUTER  :2216ROUTER       - IBM 2216 Router
```

r11.6 SP1 $TEMPLAT 0002 Definition Changes

The following list shows the new, changed, or deleted definitions for r11.6 SP1:

New

```
EE      :EE               - Enterprise Extender
```

Changed

No definitions have been changed.

Deleted

No definitions have been deleted.

r11.6 $TEMPLAT 0002 Definition Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

```
ASMON   :TNSERVER         - IBM Telnet Server
STACK   :IBMV6            - TCP/IP CS enabled for IPv6
```

Changed

```
ASMON   :STC              - Address Space Monitor
CDMGR   :CDJOB            - CONNECT:Direct JOB
CDMGR   :CDNT             - NT CONNECT:Direct Manager
CDMGR   :CDOS400          - OS/400 CONNECT:Direct Manager
CDMGR   :CDSTC            - CONNECT:Direct Started Task
CDMGR   :CDTANDEM         - TANDEM CONNECT:Direct Manager
CDMGR   :CDUNIX           - UNIX CONNECT:Direct Manager
```

```
CDMON   :ACTIVEQ          - C:D Active Queue Monitor
CDMON   :BADQ             - C:D Bad Queue Monitor
CDMON   :CALLQ            - C:D Call Queue Monitor
CDMON   :CONNECTIONS      - C:D TCP/IP Connection Monitor
CDMON   :EXECQ            - C:D Exec Queue Monitor
CDMON   :HOLDQ            - C:D Hold Queue Monitor
CDMON   :HOLDQWC          - C:D Hold Queue with WC Status
CDMON   :INITIALQ         - C:D Initial Queue Monitor
CDMON   :LISTENER         - C:D TCP/IP Listener Task Mon.
CDMON   :REMOTENODE       - C:D Remote Node Monitor
CDMON   :RETAINQ          - C:D Retain Queue Monitor
CDMON   :STARTQ           - C:D Start Queue Monitor
CDMON   :SUSPENDQ         - C:D Suspend Queue Monitor
CDMON   :TIMERQ           - C:D Timer Queue Monitor
CDMON   :TIMERQRETRY      - C:D Timer Queue - Retry Status
CDMON   :TRANSFER         - C:D Transfer Monitor
CDMON   :WAITQ            - C:D Wait Queue Monitor
CDMON   :XRTQ             - C:D Retain-Timer Queue Monitor
CICMON  :CICS             - CICS Monitor
CIP     :CIP              - Cisco Channel Intf Processor
CMMGR   :CMSTC            - C:Mailbox Started Task
CMMON   :ACQUEUE          - Queued Auto Connect Monitor
CMMON   :BSCLINES         - BSC Line Monitor
CMMON   :SNASESS          - Stalled SNA Session Monitor
CSM     :CSM              - Communications Storage Manager
EE      :EE               - Enterprise Extender
FTPMGR  :AXSJOB           - FTP TCPAccess Server - JOB
FTPMGR  :AXSSTC           - FTP TCPAccess Server - STC
FTPMGR  :AXS52JOB         - FTP TCPAccess V5.2 Server  JOB
FTPMGR  :AXS52STC         - FTP TCPAccess V5.2 Server  STC
FTPMGR  :AXS53JOB         - FTP TCPAccess V5.3 Server  JOB
FTPMGR  :AXS53STC         - FTP TCPAccess V5.3 Server  STC
FTPMGR  :CSFTPSRV         - Comms. Server FTP Server - STC
FTPMGR  :SFTPJOB          - TCPaccess FTP Server - JOB
FTPMGR  :SFTPSTC          - TCPaccess FTP Server - STC
FTPMON  :CONNECTIONS      - FTP TCP/IP Connections Monitor
FTPMON  :IPLISTEN         - FTP Listener Port Monitor
FTPMON  :REMOTENODE       - FTP Remote Node Monitor
FTSMGR  :FTSJOB           - FTS JOB
FTSMGR  :FTSREMOTE        - FTS running on a remote system
FTSMGR  :FTSSELF          - FTS running in this region
FTSMGR  :FTSSTC           - FTS Started Task
FTSMON  :LINKMONITOR      - FTS Link Monitor
INTNL   :$RMCCISR         - Monitor ISR Manager
INTNL   :$RMEV00H         - Monitor Event Handler
INTNL   :$RMIM00H         - Monitor ICON Manager
INTNL   :$RMMF00H         - Monitor SSI Message Filter
INTNL   :$RMMF01S         - Monitor Secondary AOMPROC
INTNL   :APPCLINK         - APPC Link Definition
```

```
INTNL   :NMAGENT          - Unicenter Agent
NCPMON  :NCP              - NCP template
OSA     :DIRECTNOSF       - OSA Direct (No OSASF,No SNMP)
OSA     :DIRECTXPRESS     - OSA Direct (OSASF,SNMP)
OSA     :EXPRESS          - OSA Express (OSASF,SNMP)
OSA     :EXPRESSNOSF      - OSA Express (No OSASF,No SNMP)
OSA     :OSA2             - OSA-2 (OSASF)
OSA     :OSA2NOSF         - OSA-2 (No OSASF)
ROUTER  :2216ROUTER       - IBM 2216 Router
STACK   :IBM              - TCP/IP Communications Server
STACK   :TCPACCES         - TCPaccess
VIPA    :DEFAULTDIST      - Sysplex distributor VIPA
XCMGR   :XCJOB            - XCOM as a JOB
XCMGR   :XCSTC            - XCOM Started Task
XCMON   :ACTIVE           - Active Transfer Monitor
XCMON   :CONNECTIONS      - Connections Monitor
XCMON   :HELD             - Held Transfer Monitor
XCMON   :INACTIVE         - Inactive Transfer Monitor
XCMON   :LISTENER         - Listener Monitor
XCMON   :REMOTENODE       - Remote Node Monitor
XCMON   :STALLED          - Stalled Transfer Monitor
XCMON   :SUSPENDED        - Suspended Transfer Monitor

PRCESS  :RFCDHCK1         - CDMGR Health Check - Poll C:D Stats Exit
```

Deleted

No definitions have been deleted.

$TEMPLAT 0003 Definitions

$TEMPLAT 0003 definitions have changed in r11.6.

r11.6 $TEMPLAT 0003 Definitions

The following list shows the new, changed, or deleted definitions for r11.6:

New

```
MODEL   :ADJCP          - Adjacent Control Point
MODEL   :APPNLU         - LU involved in APPN session
MODEL   :DLURPU         - Dependent LU Requestor PU
MODEL   :RTPCPCP        - Rapid Transport Protocol CPCP
MODEL   :RTPLULU        - Rapid Transport Protocol LULU
MODEL   :RTPRSTP        - Rapid Transport Protocol RSTP
MODEL   :TGPU           - Transmission Group PU
MODEL   :TRLE           - Transport Resource List Entry
```

Changed

```
INTNL   :$RMCCISR       - Monitor ISR Manager
INTNL   :$RMEV00H       - Monitor Event Handler
INTNL   :$RMIM00H       - Monitor ICON Manager
INTNL   :$RMMF00H       - Monitor SSI Message Filter
INTNL   :$RMMF01S       - Monitor Secondary AOMPROC
INTNL   :APPCLINK       - APPC Link Definition
INTNL   :NMAGENT        - Unicenter Agent
MODEL   :SWPU           - SNA Switched Physical Unit
```

Deleted

No definitions have been deleted.

Commands

Command definitions have changed in r12, r11.6 SP1, and r11.6.

Note: ALL indicates all classes or all classes except for an excluded set. INCLUDE includes a selected set of classes. NONE indicates a primary command.

r12 Command Changes

The following list shows the new, changed, or deleted definitions for r12:

New

```
APPNHPR          :D              - Display VTAM Information
APPNHPR          :S              - List RTP Pipes
STACK            :RI             - List Remote IP Addresses
STACK            :SS             - Display SSL/TLS Summary
STACK            :TC             - List TCP Application Activity
STACK(IBM)       :IF             - List FTP Connections
STACK(IBM)       :IL             - List TCP Listeners
STACK(IBM)       :IS             - Display IPSec Performance History
STACK(IBM)       :ISD            - List Dynamic Tunnels (IPSec)
STACK(IBM)       :ISF            - List IP Filters (IPSec)
STACK(IBM)       :ISK            - List IKE Tunnels (IPSec)
STACK(IBM)       :ISM            - List Manual Tunnels (IPSec)
STACK(IBM)       :ISS            - Display IPSec Summary
STACK(IBM)       :IST            - IPSec Traffic Test
STACK(IBM)       :SA             - Display AT-TLS Summary
STACK(IBM)       :SF             - Display FTP Summary
STACK(IBM)       :ST             - Display Telnet Summary

INCLUDE          :DT             - Graph TCP Connection Duration Times
INCLUDE          :IL             - List TCP Listeners
```

Changed

No definitions have been changed.

Deleted

```
ROUTER           :CI             - Display Channel Information
ROUTER           :D              - Display General Information
ROUTER           :H              - Display Performance History
ROUTER           :HI             - Display Host Interface List
ROUTER           :LU             - Display TN3270 LU Information
ROUTER           :MIB            - Display MIBinsight Browser
ROUTER           :PU             - Display TN3270 PU Information
ROUTER           :S              - Display General Information
ROUTER           :SI             - Display Host System Information
ROUTER           :TNI            - Display TN3270 Server Information
```

r11.6 SP1 Command Changes

The following list shows the new, changed, or deleted definitions for r11.6 SP1:

New

EE	:CT	- EE Connectivity Test
EE	:R	- List EE RTP Pipes
APPNHPR	:R	- List RTP Pipes
INCLUDE	:RH	- RTP Health Check
EE	:ST	- Display EE Stack Throughput
EE	:TA	- EE Traffic Analysis
EE	:TRP	- Real-Time EE Traffic by CP and
EE	:TRS	- Real-Time EE Traffic, by CP
EE	:UC	- Display UDP Connections
EE	:V	- EE VTAM Command List
APPNHPR	:V	- RTP VTAM Command List

Changed

ALL(X)	:A	- Activate a Resource
INCLUDE	:AA	- Activate Alerting
ALL(X)	:AL	- Display Alerts for a Resource
ALL(X)	:ALH	- Display Alert History for a Re
INCLUDE	:ALH	- Display Alert History for a Re
INCLUDE	:AM	- Activate Monitoring
ALL(X)	:B	- Browse Resource Status and Mod
INCLUDE	:B	- Browse Resource Monitoring Status
EE	:D	- Display Enterprise Extender
INCLUDE	:IA	- Inactivate Alerting
INCLUDE	:IC	- List IP Connections
INCLUDE	:ICF	- List IP Connections (Advanced)
INCLUDE	:IM	- Inactivate Monitoring
ALL(X)	:MA	- Set Mode Override to Automated
ALL(X)	:MAP	- Update Availability/Monitoring
ALL(X)	:MM	- Set Mode Override to Manual
ALL(X)	:MPO	- Set Map Override for a Resource
ALL(X)	:MPR	- Reset Map Override for a Resource
EE	:N	- Display SNA Node Diagnostics
INCLUDE	:PTV	- View Packet Trace
INCLUDE	:RA	- Reset Alerting Override
INCLUDE	:RM	- Reset Monitoring Activity Over
ALL(X)	:S	- Update Resource Status and Mod
EE	:S	- Display XCA Major Node Summary
ALL(X)	:T	- Terminate Resource
ALL(X)	:TF	- Force Terminate Resource
ALL(X)	:TMA	- Set Mode Override to Automated
ALL(X)	:TMI	- Set Mode Override to Ignored
ALL(X)	:TMM	- Set Mode Override to Manual
ALL(X)	:TMR	- Reset Mode Override for a Tree
INCLUDE	:UM	- Update Resource Monitoring Definition

Deleted

```
INCLUDE          :AL              - Display Alerts for a Resource
EE               :UA              - Display UDP Port Activity
```

r11.6 Command Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

```
ASMON(TNSERVER)  :CL              - Check Telnet LUs
ASMON(TNSERVER)  :CMD             - Issue System Commands to Telnet Server
ASMON(TNSERVER)  :IT              - List Telnet Connections
ASMON(TNSERVER)  :ITF             - List Telnet Connections (Advanced)
ASMON(TNSERVER)  :TWL             - Display Telnet Workload
MODEL            :RTS             - Path Switch to best route for this RTP
MODEL            :RTT             - HPR route test for this RTP PU
MODEL            :TUA             - Refresh a TRLE in TRL major node

INCLUDE          :MVT             - Move Resource Tree to Another System
INCLUDE          :RTM             - Resource Tree Monitor
NONE             :SETTLOG         - Set TLog Size
```

Changed

```
EE               :D               - Display XCA Major Node Summary
EE               :N               - Display XCA Major Node Information
EE               :SL              - Display Session List
IPNODE           :P               - PING an IP Node
IPNODE           :TR              - TRACE ROUTE for an IP Node

ALL(X)           :SVQ             - Display Services Resource Belongs To
ALL(X)           :TMA             - Set Mode Override to Automated for Tree
ALL(X)           :TMI             - Set Mode Override to Ignored for a Tree
ALL(X)           :TMM             - Set Mode Override to Manual for a Tree
ALL(X)           :TMR             - Reset Mode Override for a Tree
INCLUDE          :MV              - Move Resource Control to Another System
```

Deleted

No definitions have been deleted.

File Transfer Rules

File transfer rule definitions have changed in r11.6.

r11.6 File Transfer Rule Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

```
SAMPLE.FTPER451    - Alert on C/S FTP aborted transfers
```

Changed

No definitions have been changed.

Deleted

No definitions have been deleted.

Global Processes

Global process definitions have changed in r11.6.

r11.6 Global Process Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

No definitions have been added.

Changed

```
GENALERT           - SAMPLE: Generate an alert. Supply DESC= SEV=
```

Deleted

No definitions have been deleted.

IP Node Monitor Groups

IP node monitor group definitions have changed in r11.6.

r11.6 IP Node Monitor Group Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

```
CISCOMONINTENS    - CISCO Router I/F Errors,CPU,Mem
CISCOPERFINTENS   - CISCO Router Interface + CPU
LOWLEVEL          - Network Status hourly
STANDARD          - Network Status 10min + Alert
```

Changed

No definitions have been changed.

Deleted

No definitions have been deleted.

Macros

Macro definitions have changed in r11.6.

r11.6 Macro Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

```
GLBLSAVE          - Preserve Global Variable between restarts
```

Changed

No definitions have been changed.

Deleted

No definitions have been deleted.

Monitoring Attributes

Monitoring attribute definitions have changed in r12, r11.7, r11.6 SP1, and r11.6.

r12 Monitoring Attribute Changes

The following list shows the new, changed, or deleted definitions for r12:

New

```
DYNTUNNELACTFAILE - Dynamic tunnels failing activation
DYNTUNNELACTIVE   - Active dynamic tunnels
DYNTUNNELACTOK    - Dynamic tunnels activated successfully
DYNTUNNELBYTESIN  - Bytes in on all dynamic tunnels
DYNTUNNELBYTESOUT - Bytes out on all dynamic tunnels
DYNTUNNELEXPIRED  - Expired dynamic tunnels
DYNTUNNELINPROGRE - Dynamic tunnels with activation in progress
EEBYTESPRIHIGHRCB - Number of EE high priority bytes recvd by CP
EEBYTESPRIHIGHSEB - Number of EE high priority bytes sent by CP
EEBYTESPRILOWRCBY - Number of EE low priority bytes recvd by CP
EEBYTESPRILOWSEBY - Number of EE low priority bytes sent by CP
EEBYTESPRIMEDRCBY - Number of EE med priority bytes recvd by CP
EEBYTESPRIMEDSEBY - Number of EE med priority bytes sent by CP
EEBYTESPRINETRCBY - Number of EE net priority bytes recvd by CP
EEBYTESPRINETSEBY - Number of EE net priority bytes sent by CP
EEBYTESPRISIGRCBY - Number of EE signal bytes recvd by CP
EEBYTESPRISIGSEBY - Number of EE signal bytes sent by CP
HARDWARESTATE     - OSA Hardware State
IKEKEYMSGFAILEDAU - Key messages failing auth on IKE tunnels
IKEKEYMSGINVALID  - Invalid key messages on IKE tunnels
IKETUNNELACTIVE   - Active IKE tunnels
IKETUNNELBYTESIN  - Bytes in on all IKE tunnels
IKETUNNELBYTESOUT - Bytes out on all IKE tunnels
IKETUNNELEXPIRED  - Expired IKE tunnels
IKETUNNELINPROGRE - IKE tunnels with activation in progress
IKETUNNELLCLACTFA - IKE tunnels failing activation locally
IKETUNNELLCLACTOK - IKE tunnels activated successfully locally
IKETUNNELRMTACTFA - IKE tunnels failing activation remotely
IKETUNNELRMTACTOK - IKE tunnels activated successfully remotely
IPFILTPKTSDENIED  - Packets denied by IP filtering DENY action
IPFILTPKTSDENIEDM - Packets denied due to IP Filter mismatch
IPFILTPKTSMATCHED - Packets matching an IP Filter
OSA3DEFERREDEVENT - OSA3 deferred event count
OSA3FRAMERCVNOBUF - OSA3 frames received when no buffer
OSA3MISSEDPACKETS - OSA3 missed packets
```

Changed

No definitions have been changed.

Deleted

```
CARDSTATUS        - Oper status of adapter card
CPULOAD           - CPU Load 1 min avg
ESCONLINKSTATE    - Oper status of an Escon station
ESCONPACKETSIN    - Total inbound packets, for an Escon station
ESCONPACKETSOUT   - Total outbound packets, for an Escon station
PUSTATUS          - Oper status of tn3270 PU
TN3270BYTESIN     - Total bytes received by TN3270 server
TN3270BYTESOUT    - Total bytes sent by TN3270 server
TN3270CONREJECT   - Connections rejected
TN3270LUS         - LUs currently in use
TN3270LUSSPARE    - LUs available for use
```

r11.7 Monitoring Attribute Changes

The following list shows the new and deleted definitions for r11.7:

New

```
CHPSTATUS         - CHPID status
```

Deleted

```
HIGHPRIORITYDATA  - High priority data byte count
LOWPRIORITYDATA   - Low priority data byte count
MEDIUMPRIORITYDAT - Medium priority data byte count
NETWORKPRIORITYDA - Network priority data byte count
SIGNALTRAFFIC     - Signal traffic byte count
```

r11.6 SP1 Monitoring Attribute Changes

The following list shows the new, changed, or deleted definitions for r11.6 SP1:

New

```
EEBYTES           - Total data byte count
EEBYTESBYCP       - Total data byte count by CP
EEBYTESBYVIPA     - Total data byte count by VIPA
EEBYTESPRIHIGH    - High priority data byte count
EEBYTESPRIHIGHBYV - High priority data byte count
EEBYTESPRILOW     - Low priority data byte count
EEBYTESPRILOWBYVI - Low priority data byte count b
EEBYTESPRIMEDBYVI - Medium priority data byte count
EEBYTESPRIMEDIUM  - Medium priority data byte count
EEBYTESPRINETBYVI - Network priority data byte count
EEBYTESPRINETWORK - Network priority data byte count
EEBYTESPRISIGBYVI - Signal traffic byte count by VIPA
EEBYTESPRISIGNAL  - Signal traffic byte count
EEBYTESRECV       - Number of EE bytes received
EEBYTESRECV%STACK - EE bytes received as % of Stack
```

```
EEBYTESRECVBYCP   - Number of EE bytes received by CP
EEBYTESRECVBYVIPA - Number of EE bytes received by VIPA
EEBYTESRECVSEC    - EE Receive rate, bytes per sec
EEBYTESRECVSECBYC - EE Receive rate, bytes per sec
EEBYTESRECVSECBYV - EE Receive rate, bytes per sec
EEBYTESSEND       - Number of EE bytes sent
EEBYTESSEND%STACK - EE bytes sent as % of Stack
EEBYTESSENDBYCP   - Number of EE bytes sent by CP
EEBYTESSENDBYVIPA - Number of EE bytes sent by VIPA
EEBYTESSENDSEC    - EE Send rate, bytes per second
EEBYTESSENDSECBYC - EE Send rate, bytes per second
EEBYTESSENDSECBYV - EE Send rate, bytes per second
EECONNECTIONS     - Number of EE connections
EECONNECTIONSBYVI - Number of EE connections by VIA
EEPKTSRECV        - Number of packets received
EEPKTSRECVBYCP    - Number of packets received by CP
EEPKTSRECVBYVIPA  - Number of packets received by VIPA
EEPKTSREXMIT      - Number of packets retransmitted
EEPKTSREXMIT%     - % all packets sent that were retransmitted
EEPKTSREXMIT%BYCP - % all packets sent retransmitted
EEPKTSREXMITBYCP  - Number of packets retransmitted
EEPKTSSEND        - Number of packets sent
EEPKTSSENDBYCP    - Number of packets sent by CP
EEPKTSSENDBYVIPA  - Number of packets sent by VIPA
EERTPLUSESSBYCP   - EE RTP Pipes LU-LU sessions by
EERTPLUSESSBYVIPA - EE RTP Pipes LU-LU sessions by
EERTPLUSESSIONS   - EE RTP Pipes LU-LU sessions
EERTPPIPES        - EE RTP Pipes total
EERTPPIPESBYCP    - EE RTP Pipes total by CP
EERTPPIPESBYVIPA  - EE RTP Pipes total by VIPA
EERTPSARBRED      - EE RTP Pipes ARBmode=RED
EERTPSOVRINQLIM   - EE RTP Pipes with inbound queue
EERTPSOVROUTQLIM  - EE RTP Pipes with outbound queue
EERTPSSTALLED     - EE RTP Pipes with STALLED indicated
EERTPSSWITCHED    - EE RTP Pipes > 1 recent serious
LISTENERSEF       - TCP listener Server Efficiency Fraction
RTPLUSESSIONS     - RTP Pipes LU-LU sessions
RTPPIPES          - RTP Pipes total
RTPSARBRED        - RTP Pipes ARBmode=RED for > 5
RTPSOVRINQLIM     - RTP Pipes with inbound queues
RTPSOVROUTQLIM    - RTP Pipes with outbound queues
RTPSSTALLED       - RTP Pipes with STALLED indicator
RTPSSWITCHED      - RTP Pipes > 1 recent serious path switch
```

Changed

No definitions have been changed.

Deleted

No definitions have been deleted.

r11.6 Monitoring Attribute Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

```
ADDRESSSPACESTATU - Status of address space: active or inactive
BACKLOGBYPORT     - Connections in backlog
BACKLOGREJECTSBYP - Connections rejected due to backlog exceeded
DISTRIBUTEDABNORM - Abnormal transaction completions by port-XCF
DISTRIBUTEDCER    - Connection Establishment Rate by port-XCFadd
DISTRIBUTEDHEALTH - Health Indicator value by port-XCFaddr
DISTRIBUTEDSEF    - Server Efficiency Fraction by port-XCFaddr
DISTRIBUTEDTCSR   - Target Connectivity Success Rate by port-XCF
DISTRIBUTEDTSR    - Target Server Responsiveness by port-XCFaddr
DISTRIBUTEDWLM    - Workload Manager weight value by port-XCFadd
IFINBANDWIDTH%    - Inbound byte rate as a % of bandwidth
IFOUTBANDWIDTH%   - Outbound byte rate as a % of bandwidth
IFOUTPKTSTCPSEGGE - Outbound TCP pkts generated by seg offload
IFOUTPKTSTCPSEGOF - Outbound TCP pkts requesting seg offload
IFSTATUSMATCH     - Interface desired status matches actual
IFTOTALPKTSDISCAR - In + out packets discarded as a % of total
IFTOTALPKTSERROR% - In + out packets in error as a % of total
IPDISCARD%        - IP in packets discarded as a % of received
IPERROR%          - IP in packets in error as a % of received
IPFRAG%           - IP packets fragmented as a % of total out
IPFRAGFAILED%     - IP packets failing frag as a % of total frag
IPINBYTES         - IP bytes received including in error
IPOUTBYTES        - IP out bytes delivered for transmission
IPOUTDISCARD%     - IP out packets discarded as a % of requested
IPREASMFAILED%    - IP reassembly failures as % of reasm reqd
IPREASMREQUIRED%  - IP fragments requiring reasm as a % of rcvd
IPTRUNCATEDPKTS   - IP in packets discarded as incomplete
IP6ADDRERRORS     - IPv6 in packets with address errors
IP6DELIVERS       - IPv6 in packets delivered
IP6DGRMSFORWARDED - IPv6 in packets forwarded
IP6DGRMSUNKNWNPRO - IPv6 in packets for unknown protocol
IP6DISCARD%       - IPv6 in packets discarded as a % of received
IP6DISCARDS       - IPv6 in packets discarded
IP6ERROR%         - IPv6 in packets in error as a % of received
IP6FRAG%          - IPv6 Packets fragmented as a % of total out
IP6FRAGCREATES    - IPv6 packet fragments generated
IP6FRAGFAILED     - IPv6 packets failing fragmentation
IP6FRAGFAILED%    - IPv6 packets failing frag as a % of tot frag
IP6FRAGOK         - IPv6 Packets successfully fragmented
IP6HEADERERRORS   - IPv6 in packets with header errors
IP6INBYTES        - IPv6 bytes received including in error
IP6OUTBYTES       - IPv6 out bytes delivered for transmission
IP6OUTDISCARD%    - IPv6 out packets discarded as a % of total
IP6OUTDISCARDS    - IPv6 Out packets discarded
```

```
IP6OUTNOROUTES    - IPv6 out packets with no route
IP6OUTREQUESTS    - IPv6 out packets requested
IP6REASMFAILED    - IPv6 reassembly failures
IP6REASMFAILED%   - IPv6 reassembly failures as % of reasm reqd
IP6REASMOK        - IPv6 packets successfully reassembled
IP6REASMREQUIRED  - IPv6 fragments requiring reassembly
IP6REASMREQUIRED% - IPv6 fragments requiring reasm as % of rcvd
IP6RECEIVES       - IPv6 in packets received including in error
IP6TRUNCATEDPKTS  - IPv6 in packets discarded as incomplete
LISTENERCOUNT     - Number of TCP listeners by port
TCPDROPPEDCONNS   - TCP connections dropped
TCPINOUTOFORDER   - TCP segments rcvd out of order
TCPINOUTOFORDER%  - TCP segments rcvd out of order as % of rcvd
TCPSEGMENTSINERR% - TCP segments rcvd in error as % of rcvd
TCPSEGMENTSRXMIT% - % of TCP segments sent that were retransmits
UDPDGRMSDISCARD%  - UDP datagrams discarded as a % of rcvd
UDPDGRMSINERROR%  - UDP datagrams rcvd in error as a % of rcvd
```

Changed

```
HIGHPRIORITYDATA  - High priority data byte count
IFSTATUSOPER      - Interface actual status
IPADDRERRORS      - IP in packets with address errors
IPDELIVERS        - IP in packets delivered
IPDGRMSFORWARDED  - IP in packets forwarded
IPDGRMSUNKNWNPRO  - IP in packets for unknown protocol
IPDISCARDS        - IP in packets discarded
IPFRAGCREATES     - IP packet fragments generated
IPFRAGFAILED      - IP packets failing fragmentation
IPFRAGOK          - IP packets successfully fragmented
IPHEADERERRORS    - IP in packets with header errors
IPOUTDISCARDS     - IP out packets discarded
IPOUTNOROUTES     - IP out packets with no route
IPOUTREQUESTS     - IP out packets requested
IPREASMFAILED     - IP reassembly failures
IPREASMOK         - IP packets successfully reassembled
IPREASMREQUIRED   - IP fragments requiring reassembly
IPRECEIVES        - IP in packets received including in error
IPROUTDISCARDS    - IP routing entries discarded
LOWPRIORITYDATA   - Low priority data byte count
MEDIUMPRIORITYDAT - Medium priority data byte count
NETWORKPRIORITYDA - Network priority data byte count
SIGNALTRAFFIC     - Signal traffic byte count
TCPACTIVEOPENS    - TCP active opens
TCPCURRENTCONNS   - TCP current connections
TCPFAILEDOPENS    - TCP failed opens
TCPPASSIVEOPENS   - TCP passive opens
TCPRESETCONNECTS  - TCP connections reset
TCPSEGMENTSINERR  - TCP segments rcvd in error
TCPSEGMENTSRECVD  - TCP segments rcvd
TCPSEGMENTSRST    - TCP segments sent with RST flag
TCPSEGMENTSRXMIT  - TCP segments retransmitted
TCPSEGMENTSSENT   - TCP segments sent
UDPDGRMSINERROR   - UDP datagrams received in error
UDPDGRMSNOPORT    - UDP datagrams received for no application
UDPDGRMSRECEIVED  - UDP datagrams received
UDPDGRMSSENT      - UDP datagrams sent
```

Deleted

```
CONPACKETS        - Packets transferred for application
CONPACKETSBYIF    - Packets transferred for stack home address
CONPACKETSBYNET   - Packets transferred for network
CONTOTALPACKETS   - Packets transferred for stack
```

Prompt Lists

Prompt list definitions have changed in r12 and r11.6.

r12 Prompt List Changes

The following list shows the new, changed, or deleted definitions for r12:

New

```
FT43DB2          - ASMON Type (DB2) - Force INACT Message
IN43DB2          - ASMON Type (DB2) - Activation Message
MO43DB2          - ASMON Type (DB2) - Status Monitor Message
TR43DB2          - ASMON Type (DB2) - INActivation Message
```

Changed

```
$RM43TYPE        - ASMON Types
```

Deleted

No definitions have been deleted.

r11.6 Prompt List Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

```
$RM$RAMDBLVL     - A record to hold the RAMDB maintenance level
$RM55GRPTYPE     - SNA Group Resource Model Type
FT02MQ           - Started Task Type (MQ) - Force INACT Message
FT43TNSERVER     - ASMON Type (TNSERVER) - Force INACT Message
IN02MQ           - Started Task Type (MQ) - Activation Message
IN43TNSERVER     - ASMON Type (TNSERVER) - Activation Message
MO02MQ           - Started Task Type (MQ) - Status Monitor Message
MO43TNSERVER     - ASMON Type (TNSERVER) - Status Monitor Message
TR02MQ           - Started Task Type (MQ) - INActivation Message
TR43TNSERVER     - ASMON Type (TNSERVER) - INActivation Message
```

Changed

```
$RMAMMSORT       - Alert Monitor Sort
$RM00TEMPLAT     - Template Name
$RM40CMDUSER     - Command Execution User ID
$RM43TYPE        - ASMON Types
$RM55TYPE        - SNA Resource Model Type
```

Deleted

No definitions have been deleted.

Alert Monitor Filters

Status monitor filter definitions have changed in r12.

r12 Alert Monitor Filter Changes

The following list shows the new, changed, or deleted definitions for r12:

New

```
CONNSTAT          - IP Event Detector CONNSTAT (active connection count)
FRAGMENT          - IP Event Detector FRAGMENT (IP packet fragmentation)
SSLHFAIL          - IP Event Detector SSLHFAIL (SSL handshake failure)
SVRRESET          - IP Event Detector SVRRESET (TCP server reset)
TCPSTART          - IP Event Detector TCPSTART (TCP connection start)
TCPSTOP           - IP Event Detector TCPSTOP (TCP connection stop)
```

Changed

```
CIP               - All CIP Resources
IPALL             - All IP Related Resources (Includes IPNODES)
IPRSC             - All IP Related Resources (Excludes IPNODES)
```

Deleted

No definitions have been deleted.

Status Monitor Filters

Status monitor filter definitions have changed in r12, r11.6 SP1, and r11.6.

r12 Status Monitor Filter Changes

The following list shows the new, changed, or deleted definitions for r12:

New

```
DB2               - All ASMON resources of type DB2
```

Changed

```
CIP               - All CIP Resources
IPALL             - All IP Related Resources (Includes IPNODES)
IPRSC             - All IP Related Resources (Excludes IPNODES)
```

Deleted

```
CIPRTR            - All CIP and Router Resources
ROUTER            - All ROUTER Resources
```

r11.6 SP1 Status Monitor Filter Changes

The following list shows the new, changed, or deleted definitions for r11.6 SP1:

New

```
APPNHPR          - All APPNHPR Resources
```

Changed

```
IPALL            - All IP Related Resources (Include)
IPRSC            - All IP Related Resources (Exclude)
```

Deleted

No definitions have been deleted.

r11.6 Status Monitor Filter Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

```
CIPRTR           - All CIP and Router Resources
TNSERVER         - All ASMON Resources of Type TNSERVER
```

Changed

```
ALERTS           - Show resources with Alert Count > 0
```

Deleted

No definitions have been deleted.

VTAM States

VTAM state definitions have changed in r11.6.

r11.6 VTAM State Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

```
PGAIN                    - Pending EE Hostname Resolution
```

Changed

No definitions have been changed.

Deleted

No definitions have been deleted.

Resource Groups

Resource group definitions have changed in r12.

r12 Resource Group Changes

The following list shows the new, changed, or deleted definitions for r12:

New

No definitions have been added.

Changed

No definitions have been changed.

Deleted

```
CIPRT                    - CIPs and ROUTERs
ROUTR                    - ROUTERS
```

Icon Panels

Icon panel definitions have changed in r12, r11.6 SP1, and r11.6.

r12 Icon Panel Changes

The following list shows the new, changed, or deleted definitions for r12:

New

No definitions have been added.

Changed

```
$IPDEFAULT        - Default Graphic Monitor - TCPIP
```

Deleted

No definitions have been deleted.

r11.6 SP1 Icon Panel Changes

The following list shows the new, changed, or deleted definitions for r11.6 SP1:

New

```
$ICEEHPR                  - EE and APPN/HPR
EEHPR                     - EE and APPN/HPR
```

Changed

```
$IPDEFAULT        - Default Graphic Monitor - TCPIP
ICONCOUNTS        - Resource Counts for Icon Display
```

Deleted

No definitions have been deleted.

r11.6 Icon Panel Changes

The following list shows the new, changed, or deleted definitions for r11.6:

New

No definitions have been added.

Changed

```
$IPDEFAULT        - Default Graphic Monitor - TCPIP
```

Deleted

No definitions have been deleted.

Index

$

$CMDENT • 49
$NDJPROC procedure • 272
$NMLOCK procedure • 273
$NMPMENU procedure • 274
$RECALL API • 322
$RECALL API, EventView variables
  retrieving the value of • 360
  setting the value of • 358
$REDBAPI API • 323
$REDBAPI API, EventView definitions
  creating • 363
  deleting • 370
  listing • 374
  retrieving information abo • 372
$REDBAPI API, field names
  EventView message rules • 368
  EventView rule sets • 367
$RMCALL API • 319
  Automation Services commands, executing • 324
  knowledge base, retrieving information from • 329
  link records, deleting • 331
  resource status information, retrieving • 332
$RMDBAPI API
  syntax • 320
$RMDBAPI API, field names
  resources • 340
  system images • 339
$RMDBAPI API, ResourceView definitions
  creating • 334
  deleting • 347
  listing • 352
  retrieving information • 349
$RMDBAPI API, system image definitions
  field values, changing • 354
$RMEVENT API • 321, 356
$RMSTSET API • 321, 357

&

&CALL
  performance considerations • 49
  sharing subpool zero • 264
&CONTROL RESCAN • 49
&DOM
  statement • 74
  verb • 75
&INTCMD verb • 220
&INTREAD verb • 220
&RETCODE verb • 75
&SMFWRITE
  authorization exit • 264
  statement • 264
  verb • 264
&USERPW system variable • 293
&WTO
  statement • 72
  verb • 75
&WTOR
  statement • 73
  verb • 75
&ZDOMID
  system variable • 74
  verb • 75
&ZFDBK verb • 75
&ZMAOMAU system variable • 75
&ZMAOMBC verb • 75
&ZMAOMDTA verb • 75
&ZMAOMID verb • 75
&ZMAOMJI verb • 75
&ZMAOMJN verb • 75
&ZMAOMMID verb • 75
&ZMAOMMIN verb • 75
&ZMAOMMLC verb • 75
&ZMAOMMLD verb • 75
&ZMAOMMLE verb • 75
&ZMAOMMLL verb • 75
&ZMAOMMLT verb • 75
&ZMAOMMLV verb • 75
&ZMAOMMSG verb • 75
&ZMAOMRC verb • 75
&ZMAOMRCM verb • 75
&ZMAOMRCX verb • 75
&ZMAOMSOS verb • 75
&ZMAOMTM verb • 75
&ZMAOMTYP verb • 75
&ZMAOMUFM verb • 75
&ZMPTEXT system variable • 71

A

ACBRETRY operand • 264
ACBs
  automating control of • 117
activation
  preprocessing • 133
  process return codes • 118
Activation Details panel • 118
activity logs
  define lines per page • 264
  specify procedure • 264
alerts
  VSAM • 43
AOM START command • 75
AOM STOP command • 75
AOMCUTOK operand • 264
AOMMLTO operand • 264
AOMPRFJI operand • 264
AOMPRFJN operand • 264
AOMPRFMN operand • 264
AOMPRFSN operand • 264
AOMPRFTM operand • 264
AOMPROC procedure • 228
AOMSSID operand • 264
AOMTRACE operand • 264
AOMTRCRC operand • 264
AOMTRLIM operand • 264
API (application program interface)
  procedures • 318
APIs, specific
  $RECALL • 322, 358
  $REDBAPI • 323
  $RMCALL • 319
  $RMDBAPI • 320
  $RMEVENT • 321, 356
  $RMSTSET • 321, 357
APPC links • 229
  display status • 230
  start • 229
  stop • 231
ARMNAME product region JCL parameter • 247
AUTOEXEC operand • 264
Automated Mode Attributes Table • 182
Automation Services
  execute commands from an external source • 324
  macro template • 167
  macros • 167
Automation Services commands
  access definitions • 172
  what they are • 171
availability map definitions, adding • 118
Availability Map panel • 118

B

BIND parameters • 102
BIND RUSIZE • 46
Broadcast Services
  customizing requirements • 233
buffer sharing • 40

C

CA SOLVE:FTS
  traffic • 202
  tune • 48
CALLSHR0 operand • 267
CDELAY operand • 267
centralized control, remote regions
  with ISR • 224
  with ROF • 219
CHECKALL command and Display Attribute Tables • 180
CICS (Customer Information Control System)
  consoles for • 315
  MAI-OC, with • 88
CISIZE • 42
CMDREPLS parameter group • 23
CNMACBNM operand • 267
CNMPROC procedure
  ISR, and • 228
color, displayed status • 181
column width, status monitor • 191
command authority, change • 264
command NCL procedures, variables • 176
command stack size • 264
commands
  add definitions • 175
  define your own • 175
  delete definitions • 175
  execute using $RMCALL API • 324
  for NETMASTR • 31
  for performance and tuning • 50
  maintain definitions • 175
  register • 171
commands, specific
  AOM START • 75
  AOM STOP • 75
  CHECKALL • 180, 324

DD • 32
EQUATE • 23
ERROR • 31
EVERY • 264
GLOBAL • 327
LINK • 204
LINK RESET • 207
LINK START • 206
LINK STOP • 206
LIST • 31
LOAD • 327
LOCK • 264
PARMSEP • 31
PF • 22
PFLIST • 21
PGM • 31
PPREF • 31
PROFILE • 75, 264
PSUFF • 31
SHOW AOMABEND • 75
SHOW AOMSTAT • 75
SHOW CONSOLES • 75
SHOW ISR • 227
SHOW ISRSTATS • 48
SHOW LSR • 45
SHOW MAI • 98
SHOW NCL • 50
SHOW NDB=ALL • 46
SHOW SYSPGT • 51
SHOW SYSRCT • 53
SHOW SYSWAIT • 56
SHOW VSAM • 44
SHOW VSAMIO • 41
STATUS • 75
SUBMIT • 75
SUBS • 31
SYSCMD • 75
SYSPARMS • 71, 75
SYSPGT • 50, 58
SYSRCT • 50, 58
UDBCTL • 40
USER ACCT • 63
VAR • 31
common ISR functionality • 228
communicate between domains
INMC • 199
tuning • 46
components • 467
CONMSG operand • 264
connect
terminals to your region • 244
to a session • 97
consoles
CICS, define to • 315
enable MASTER authority • 308
extended MCS • 308
JES • 307
management • 309
number of • 310
OP1 • 307
OP2 • 307
pool • 310
pseudo • 307
types used • 307
contacting technical support • iv
control characters (SCS) • 99
conversations, ISR • 224, 226
COS definitions • 36
CPU consumption • 310
CPU-time accounting
implement • 61
starting • 63
status • 65
stop • 63
cross-domain definitions • 90
customer support, contacting • iv
customize
access to external applications • 81
display attribute tables • 179
logon scripts • 83
prompt list • 195
your region • 21
Customizer parameter groups
CMDREPLS • 23
EQUATES • 23
EXTAPPLS • 81

D

DALDEFER operand • 268
DALRACF operand • 268
DALRLSE operand • 268
data exchange, ISR
control • 228
without a procedure • 229
data set descriptions • 467
data sets
allocate dynamically • 32
types • 467

data streams
SCS control characters • 99
database
backups • 159
maintenance • 157
DBCS product region JCL parameter • 247
DD command • 32
defer data set updates • 40
definitions
production machine • 104
testing machine • 106
delay interval • 264
DESC operand • 268
devices supported • 243
diagnose problems, INMC links • 209
disconnect from a session • 97
Display and Heartbeat Details panel • 128
display attribute tables
and the CHECKALL command • 180
customize • 179
types • 179
display methods
considerations • 128
display process return codes • 128
DYNLMAX operand • 268
DYNVVOL product region JCL parameter • 247

E

EDITCAPS operand • 264
EDITMAXK operand • 264
EDITNULL operand • 264
environments using ISR • 224
environments, with ROF • 220
EQUATE command
include in INIT and READY • 220
set • 23
EQUATE values for MAI-OC commands • 97
EQUATES parameter group • 23
ERROR command • 31
EVCMDMIN operand • 264
EventView
SMF record format • 379
store statistics • 377
EventView rule sets, using an API to
create information • 363
delete information • 370
list definitions • 374
retrieve information • 372
EventView variable values
retrieve • 360, 362
setting • 358
EVERY command repeat interval • 264
exits
MAI • 98
extended attributes support • 244
Extended Function Exit panel • 136
extended MCS consoles • 308
effect on performance • 310
migration IDs • 309
SYSCMD • 69
external applications access
customize • 81
log on • 83

F

field names, $REDBAPI
EventView message rules, used in • 368
EventView rule sets, used in • 367
field names, $RMDBAPI
resources, used in • 340
system images, used in • 339
Force Inactivation Details panel • 126
forced inactivation process return codes • 126
formats
NMDRVCTL control statements • 30
free space allocation • 42
FTSCPROC operand • 264
FTSMAXBK operand • 264
FTSRCDSN operand • 264
FTSSMF operand • 264
FTSTRDSN operand • 264
function keys
assignments, controlling • 21

G

generate SMF records • 61
global function keys • 21
graphical monitor
color of displayed status • 181
display attribute table data • 181

H

held messages, specifying number • 264
HELDMSG operand • 49, 264
hide password • 101
highlight displayed status • 181

I

IMS (Information Management System)
  with MAI-OC • 88
IMSP production machine definitions • 105
IMST testing machine definitions • 107
Inactivation Details panel • 124
inactivation process return codes • 124
INIFILE product region JCL parameter • 248
INIT product region JCL parameter • 248
INIT with EQUATE command • 220
INMC links • 201
  BIND RUSIZE • 46
  buffer size • 264
  control • 206
  define • 217, 218
  delete definition • 219
  diagnose problems • 209
  improve performance • 208
  ISR traffic • 202
  logmode table • 207
  maximum number • 264
  modify definition • 218
  planning • 211
  reset • 207
  ROF traffic • 202
  rotate and backup • 211, 213
  RU size • 209
  start • 206
  static • 203
  status • 208
  stop • 206
  trace • 209
  traffic flow • 202
  traffic synchronization, preferential links • 215
  transmission buffer size • 208
  troubleshooting • 210
  tune • 46
INMC links, multipath • 201
  establish • 216
  multiple sessions • 201
INMC links, preferential • 211
  define • 218
  planning • 215
INMC links, simple-mode • 211
  plan • 211
  session parameters • 212
INMC links, specify
  COS table entry • 207
  traffic route • 207
INMCBFSZ operand • 46, 208, 264
INMCEX01 operand • 264
INMCEX02 operand • 264
installation exits • 98
INT product region JCL parameter • 248
intensity, displayed status • 181
IPAMHB operand • 264
ISR
  common functionality • 228
  conversation classes • 224
  data exchange without a procedure • 229
  environments that use • 224
  establish conversations • 226
  message loop protection • 225
  monitor status • 227
  plan • 225
  status • 227
  traffic • 202
  tune • 48
ISR, control
  data exchange • 228
  remote regions • 224

J

JCL parameters, specific
  VSAMIO • 41
JES consoles • 307
JES consoles and SYSCMD • 68
JES sessions, terminating • 102
JES2 and MAI-OC • 89, 98
JES3 and MAI-OC • 89, 98
journal swapping • 264
JRNLPROC operand • 264
JRNLSWAP operand • 264

K

keyboard lock • 98
keyboard unlock • 98
knowledge base definitions
  classes • 383

L

LANG operand • 24, 264
language code • 24, 264
LINK command • 204
LINK RESET command • 207
LINK START command • 206
LINK STOP command • 206

LIST command • 31
LMP codes • 247
LMSGWARN operand • 264
LNKTRACE operand • 209, 264
LOCATE command • 165
lock clearing • 71
LOCK command • 264
LOCKPROC operand • 264
log on to external applications • 83
logical resources, timeout on response • 137
Logical State Attributes Table, edit • 181
logical state mapping
  AUTOMATED operation mode • 183
  MANUAL operation mode • 184
logmode tables
  INMC links, and • 207
logon
  attempts, number allowed • 264
  script, customize • 83
LOGPAGE operand • 264
LSR
  buffer sharing • 40
  monitoring tuning • 45
LSRPOOL parameter group • 40
LU1
  logons, alternate menu • 274
LUs (logical units)
  MAI-OC • 87
  pool • 92
  select • 91
  specifying a terminal • 91

M

Macro Definition panel • 169
macros
  access definitions • 169
  add definitions • 169
  delete definitions • 169
  find references to • 161
  maintain definitions • 169
  OPNDST • 244
  register and maintain • 168
  search for • 163
MAIACBOR operand • 264
MAIDISC command • 102
MAIEX02 exit
  correlators • 299
  invocation • 295
  operand • 264
  reentrancy • 296
  registers on entry • 299
  sample • 298
  serialization • 297
  starting • 298
  storage subpools • 296
  SYSPARMS operand • 296
MAIEX02 exit, communications area contents • 300
  ACB open call • 303
  exit initialization call • 300
  exit termination call • 301
  MAI session end call • 304
  MAI session start call • 302
MAIEX02 exit, correlators
  security exit • 300
  session • 299
  system • 299
  user • 299
MAIEX02 exit, return codes • 304
  exit initialization • 304
  session start • 305
MAIEX02S system parameter
  operand • 264, 297
  sample exit • 298
MAI-OC • 86
  logmode entry selection • 103
  on CICS systems • 88
  on IMS systems • 88
  on JES2 systems • 89
  on JES3 systems • 89
  SNA LU-Type 1 • 98
MAIONL operand • 264
MAIOPREF operand • 264
MAIOTRNS operand • 264
Manual Mode Attributes Table • 183
MAPDEL operand • 264
MAPLOAD operand • 264
MAPRESET operand • 264
MAXRUSZ operand • 264
MCS (multiple console support), extended consoles • 308
MENULU1 operand • 264
MENUPROC operand • 264
message rules
  search for • 163
message rules, use an API to
  create information • 363
  delete information • 370

list definitions • 374
retrieve information • 372
messages
MAI-OC • 97
resource definitions, in • 139
messages, maximum number
held messages • 264
trace messages • 264
migration IDs, extended MCS consoles • 309
MODDEL operand • 264
models
user ID • 274
MODETABS member • 246
MODIFY facility • 247
MODLOAD operand • 49, 264
MODLUSER operand • 264
modules, preload • 264
MSGPROC procedure • 71
MSP environment, suitable console types • 307

N

names
product name keys • 254
National Language character set support
customize • 24
NCL panel processing • 24
related functions • 25
statements • 24
NCL (Network Control Language)
code your own command procedure • 176
variables available to command • 176
verb summary for SYSCMD • 75
writing procedures to be used a • 167
NCL authorization exit name and NCLEX01 system parameter • 264
NCL procedures
$NDJPROC • 272
$NMLOCK • 273
$NMPMENU • 274
improve procedure use • 38
monitor procedure loading • 39
preload • 39
SYSCMD • 70
unload • 264
use with ROF • 220
NCLEX01 operand • 264
NCLFM product region JCL parameter • 248
NCLGBTRC operand • 264
NCLOGTRM operand • 264
NCLTRLFF operand • 264
NCLTRMAX operand • 264
NDB, tuning • 39
database activity • 43
record sizes • 42
NDBLOGSZ operand • 264
NDBOPENX operand • 264
NDBPHONX operand • 264
NDBSCANO operand • 264
NDBSUBMN operand • 43, 264
NDBSUBMX operand • 43, 264
NETMASTR program • 29
commands • 31
process • 29
NMDID product region JCL parameter • 248
NMDRVCTL data set • 30
NMIQLIM operand • 264
NMREADY • 204
NMSUP product region JCL parameter • 248
NOMODIFY product region JCL parameter • 248
NONSWAP operand • 264
NPF product region JCL parameter • 248
NPFFM product region JCL parameter • 248
NRDLIM operand • 264
NSPRTINT operand • 264
NTSACCT operand • 264
NTSCINTV operand • 264
NTSCLOSE operand • 264
NTSCNMQ operand • 264
NTSEVENT operand • 264
NTSMAIEX operand • 276
NTSMAISV operand • 276
NTSMAXTP operand • 276
NTSMAXTR operand • 276
NTSRSINT operand • 276
NTSRSLIM operand • 276
NTSRSTAT operand • 276
NTSSAWBF operand • 276
NTSSKEEP operand • 276
NTSTRBFX operand • 276
NTSTRCBF operand • 276

O

OCS command stack size • 264
OCS commands
MAIDISC • 102
OPSYS • 102
OCSHLITE operand • 284
OCSTIME operand • 284

OP1 consoles • 307
OP2 consoles • 307
operation modes
  global • 117
  resource definition • 117
OPNDST macro • 244
OPSYS command • 102
OPT product region JCL parameter • 249
OSINP product region JCL parameter • 249

P

panels, resource definition
  Activation Details • 118
  Availability Map • 118
  Display and Heartbeat Details • 128
  Extended Function Exit • 136
  Force Inactivation Details • 126
  Inactivation Details • 124
  number held in storage • 264
  Restart Control Parameters • 124
  State Change Exits • 133
  tuning • 37
PANLBFSZ operand • 37, 284
PANLBUFF operand • 37, 284
parameter groups
  CMDREPLS • 23
  EQUATES • 23
  EXTAPPLS • 81
  LSRPOOL • 40
PARMSEP command • 31
passwords
  expiry interval • 264
  length • 264
performance and tuning commands • 50
  SHOW SYSPGT • 51
  SHOW SYSRCT • 53
  SHOW SYSWAIT • 56
  SYSPGT • 58
  SYSRCT • 58
performance enhancements • 35
  by controlling OCS message flow • 49
  by deferring VSAM data set updates • 40
  by performing VSAM I/O in subtask • 41
  by tuning communication between domains • 46
  by tuning record size • 42
  by using buffer sharing • 40
  by using COS definitions • 36
performance tuning
  at the system level • 36
  data set activity • 43
  panel use • 37
  storage limits for panel sends • 37
  VTAM interface • 36
performance, NCL procedures
  considerations when writing NCL • 49
  improve NCL procedure use • 38
  monitor NCL procedure loading activity • 39
  preload • 39
PF command • 22
PF LIST command • 21
PGM command • 31
physical terminal • 87
PPOCOLOR operand • 284
PPOHLITE operand • 284
PPOPROC
  procedure • 228
PPOSOMSG operand • 284
PPOUSMSG operand • 284
PPREF command • 31
preferential INMC links • 211
  define • 218
  plan • 215
  traffic synchronization • 215
preload NCL procedures • 39, 264
preparse facilities • 49
PRI product region JCL parameter • 250
primary menu
  alternate • 264
  procedure • 25
printing
  SCS control characters • 99
problem determination, INMC • 210
processes
  search for • 164
PROD product region JCL parameter • 250
product region JCL parameters, specific
  ARMNAME • 247
  DBCS • 247
  DYNVVOL • 247
  INIFILE • 248
  INIT • 248
  INT • 248
  NCLFM • 248
  NMDID • 248
  NMSUP • 248
  NOMODIFY • 248
  NPF • 248

NPFFM • 248
OPT • 249
OSINP • 249
PRI • 250
PROD • 250
READY • 250
SEC • 251
SSID • 251
TZ • 252
UDBDEFER • 252
VFSENQ • 252
WTO • 253
XM • 253
products
name keys • 254
prompt lists
add a value to • 196
disable variable substitution • 196
greater than signs (>) • 197
less than signs (<) • 196
maintain definitions • 198
variable substitution • 196
variables in an entry • 196
PSERVIC field, logmode table definition • 245
pseudo consoles • 307
PSUFF command • 31
PWEXPIRE operand • 284
PWMAX operand • 284
PWMIN operand • 284
PWRETRY operand • 284

Q

query, extended attributes • 244
queue limits for message traffic • 264

R

READY product region JCL parameter • 250
READY with EQUATE command • 220
RECSZ • 42
reporting
about • 161, 164
locate output • 165
macros, for • 163
message rules, for • 163
output formats • 161
processes, for • 164
search, perform • 162
sort output • 165
requests, unit size • 264
resource definition panels
Activation Details • 118
Availability Map • 118
Display and Heartbeat Details • 128
Extended Functions Exit • 136
Force Inactivation Details • 126
Inactivation Details • 124
Restart Control Parameters • 124
State Change Exits • 133
resource definitions
ACB, controlling • 117
classes • 383
messages, specifying • 139
operation mode • 117
system load balancing • 152
using timeout on response • 137
resource starting, sequence of events • 121
resources
restart control • 122
set the actual state of • 357
resources, using an API to
create information • 334
delete information • 347
list definitions • 352
retrieve information • 329, 349
retrieve status information • 332
send messages • 356
ResourceView
SMF record format • 380
statistics, storing • 377
Restart Control Parameters panel • 124
manual activation, effect on • 124
suppressing restart • 124
restart control, resources • 122
return codes
activation process • 118
display process • 128
forced inactivation process • 126
inactivation process • 124
ROF (Remote Operator Facility)
background environments, with • 220
control remote regions • 219
NCL procedures, with • 220
system console, with • 220
traffic • 202
tune • 47
user IDs • 219
rotate and backup INMC links • 211
plan • 213

traffic synchronization • 215
ROUTCDE operand • 287
RU size, increase • 209
RXQSFIX operand • 264

S

sample MAIEX02 exit • 298
screen sizes
specify defaults • 264
specify size • 245
unspecified viewport size • 245
SCS character support • 99
SEC product region JCL parameter • 251
send query to terminal • 244
ServiceView
SMF record format • 380
store statistics • 377
sessions
connection • 97
cross domain • 90
disconnection • 97
JES • 102
multiple • 201
protocols • 98
specify a terminal • 91
with other systems • 87
SESSMSG operand • 209, 264
SHOW AOMABEND command • 75
SHOW AOMSTAT command • 75
SHOW CONSOLES command • 75
SHOW ISR command • 227
SHOW ISRSTATS command • 48
SHOW LSR command • 45
SHOW MAI command • 98
SHOW NCL command • 50
SHOW NDB=ALL command • 46
SHOW SYSPGT command • 51
SHOW SYSRCT command • 53
SHOW SYSWAIT command • 56
SHOW VSAM command • 44
SHOW VSAMIO command • 41
simple-mode INMC links • 211
COS table entry • 212
plan • 211
session parameters • 212
SMF
EventView record format • 379
generate • 61
header format • 378
identifier • 264
record types • 377
ResourceView record format • 380
ServiceView record format • 380
user-defined record format • 381
SMFID operand • 264
SMFWRITE macro • 377
SNA Resource Logical State Normalization Table • 185
SORT command • 165
specify
terminals • 91
SSID product region JCL parameter • 251
State Change Exits panel • 133
static INMC links • 203
STATUS command • 75
status monitor
color of displayed status • 181
use display attribute table data • 181
views • 188
status monitor display format • 187
column width • 191
extended display • 191
headings • 191
multiscreen • 194
specify • 191
variables • 191
status of CPU-time accounting • 65
status, color • 181
STGWRN operand • 264
STGWRNXA operand • 264
storage
limits for panel sends • 37
maximum amount for editor • 264
strike-over masks • 101
SUBMIT command • 75
SUBS command • 31
support, contacting • iv
supported devices • 243
suppress restarts • 124
symbols, system • 29
SYSCMD
AOM command summary • 75
facility • 67
message prefixes • 71
NCL verbs summary • 75
variables summary • 75
SYSCMD command • 75
&INTCMD environment • 70

command summary • 75
z/OS, MSP, VOS3 • 67
SYSCONNM operand • 264
SYSCONSO operand • 264
SYSCONUI operand • 264
SYSCONXU operand • 264
SYSLOG operand • 264
SYSLOGFM operand • 264
SYSPARMS operands
ACBRETRY • 264
AOMCUTOK • 264
AOMMLTO • 264
AOMPRFJI • 264
AOMPRFJN • 264
AOMPRFMN • 264
AOMPRFSN • 264
AOMPRFTM • 264
AOMSSID • 264
AOMTRACE • 264
AOMTRCRC • 264
AOMTRLIM • 264
AUTOEXEC • 264
CALLSHR0 • 267
CDELAY • 267
CNMACBNM • 267
CONMSG • 267
DALDEFER • 268
DALRACF • 268
DALRLSE • 268
DESC • 268
descriptions • 264
DYNLMAX • 268
EDITCAPS • 268
EDITNULL • 268
EVCMDMIN • 268
FTSCPROC • 264
FTSMAXBK • 264
FTSRCDSN • 264
FTSSMF • 269
FTSTRDSN • 269
HELDMSG • 49, 271
ID • 204
INMCBFSZ • 46, 208, 271
INMCEX01 • 271
INMCEX02 • 271
IPAMHB • 271
JRNLPROC • 272
JRNLSWAP • 272
LANG • 24, 273
LMSGWARN • 273
LNKTRACE • 209, 273
LOCKPROC • 273
LOGPAGE • 273
MAIACBOR • 274
MAIEX02 • 274
MAIEX02S • 274, 297
MAIONL • 274
MAIOPREF • 274
MAIOTRNS • 274
MAXRUSZ • 274
MENULU1 • 274
MENUPROC • 274
MODLUSER • 274
NCLEX01 • 276
NCLGBTRC • 276
NCLOGTRM • 276
NCLTRLFF • 276
NCLTRMAX • 276
NDBLOGSZ • 276
NDBOPENX • 276
NDBPHONX • 276
NDBSCANO • 276
NDBSUBMN • 43, 276
NDBSUBMX • 43, 276
NONSWAP • 276
NRDLIM • 276
NSPRTINT • 276
NTSACCT • 276
NTSCINTV • 276
NTSCLOSE • 276
NTSCNMQ • 276
NTSEVENT • 276
NTSMAIEX • 276
NTSMAISV • 276
NTSMAXTP • 276
NTSMAXTR • 276
NTSRSINT • 276
NTSRSLIM • 276
NTSRSTAT • 276
NTSSAWBF • 276
NTSSKEEP • 276
NTSTRBFX • 276
NTSTRCBF • 276
OCSHLITE • 284
OCSTIME • 284
PANLBFSZ • 37, 284
PANLBUFF • 37, 284
PPOCOLOR • 284

PPOHLITE • 284
PPOSOMSG • 284
PPOUSMSG • 284
PWEXPIRE • 284
PWMAX • 284
PWMIN • 284
PWRETRY • 284
ROUTCDE • 287
SESSMSG • 209, 289
SMFID • 289
SSIRTINT • 289
STGWRN • 289
STGWRNXA • 289
summary table • 257
SYSCONNM • 289
SYSCONSO • 289
SYSCONUI • 289
SYSCONXU • 289
SYSLOG • 289
SYSLOGFM • 289
TNDSREG • 292
TRACEOPT • 292
USERPW • 293
VDISPLAY • 293
VTAMID • 293
XABELOW • 293
SYSPARMS operands, REXX
RXQSFIX • 264
SYSPGT command • 50, 58
sysplex
NETMASTR • 29
SYSRCT command • 50, 58
system console using with ROF • 220
system identifier • 264
system images
use an API to change information • 354
system load balancing • 152
system variables
&USERPW • 264
&ZDOMID • 74
&ZMAOMAU • 75
&ZMPTEXT • 71

T

technical support, contacting • iv
terminals
connect • 244
specify screen sizes • 245
time-out facility • 23
unspecified viewport size • 245
test machine definitions • 106
time-outs
actions • 264
intervals • 264
response to actions • 137
TNDSREG operand • 264
TRACEOPT operand • 264
tracing
INMC links • 209
messages, maximum number • 264
options • 264
traffic
flow, INMC • 202
synchronization, INMC • 215
transmission buffer size • 208
troubleshooting INMC • 210
types, data set • 467
TZ product region JCL parameter • 252

U

UDB, tuning • 39
UDBCTL command • 40
UDBDEFER product region JCL parameter • 252
unspecified viewport size
defined • 245
user abend code, specify • 264
USER ACCT command • 63
user data, accept at logon • 264
user exits, migration IDs • 307
user ID, define for ROF • 219
user-defined SMF record format • 381
USERPW operand • 264
UTIL0028 utility • See NETMASTR program

V

VAR command • 31
variables
&ZRMCMDPARMS • 176
command NCL procedures • 176
status monitor display format • 191
variables, EventView
retrieve the value of • 360, 362
set the value of • 358
VDISPLAY operand • 264
verbs
&DOM • 75
&INTCMD • 220
&INTREAD • 220

&RETCODE • 75
&SMFWRITE • 276
&WTO • 75
&WTOR • 75
&ZDOMID • 75
&ZFDBK • 75
&ZMAOMBC • 75
&ZMAOMDATA • 75
&ZMAOMID • 75
&ZMAOMJI • 75
&ZMAOMJN • 75
&ZMAOMMID • 75
&ZMAOMMIN • 75
&ZMAOMMLC • 75
&ZMAOMMLD • 75
&ZMAOMMLE • 75
&ZMAOMMLL • 75
&ZMAOMMLT • 75
&ZMAOMMLV • 75
&ZMAOMMSG • 75
&ZMAOMRC • 75
&ZMAOMRCM • 75
&ZMAOMRCX • 75
&ZMAOMSOS • 75
&ZMAOMTM • 75
&ZMAOMTYP • 75
&ZMAOMUFM • 75
VFS
database • 217, 218
VFSENQ product region JCL parameter • 252
views, status monitor • 188
virtual routes
INMC links • 207
VOS3 environment, suitable console types • 307
VSAM
alerts • 43
VSAM data sets
defer updates • 40
I/O in subtask • 41
share buffers • 40
VSAM data sets, tuning • 39
activity • 43
monitor • 44, 46
size • 42
VSAMIO JCL parameter • 41
VTAM
ACB, reopening • 264
APPL • 102
definitions • 104
installation • 104
logmode table, extended attributes • 244
PPO ACB name, changing • 264
sessions • 201
start • 264
VTAM testing machine definitions • 106
VTAMID operand • 264

W

warning for lost messages • 264
write NCL, performance considerations • 49
WTO product region JCL parameter • 253

X

XABELOW operand • 264
XM product region JCL parameter • 253

Z

z/OS environment, suitable console types • 307