# Unicenter

## Management Services Administrator Guide

Version 5.0

P01-045

Booklist

Computer Associates
The Software That Manages eBusiness

| Edition | Publication Number | Publish Date |
|---|---|---|
| 8th Edition | P01-045 | October 2001 |
| 8th Edition—Update 1 | P01-045 | May 2002 |



This documentation and related computer software program (hereinafter referred to as the "Documentation") is for the end user's informational purposes only and is subject to change or withdrawal by Computer Associates International, Inc. ("CA") at any time.

This documentation may not be copied, transferred, reproduced, disclosed or duplicated, in whole or in part, without the prior written consent of CA. This documentation is proprietary information of CA and protected by the copyright laws of the United States and international treaties.

Notwithstanding the foregoing, licensed users may print a reasonable number of copies of this documentation for their own internal use, provided that all CA copyright notices and legends are affixed to each reproduced copy. Only authorized employees, consultants, or agents of the user who are bound by the confidentiality provisions of the license for the software are permitted to have access to such copies.

This right to print copies is limited to the period during which the license for the product remains in full force and effect. Should the license terminate for any reason, it shall be the user's responsibility to return to CA the reproduced copies or to certify to CA that same have been 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, business interruption, goodwill, or lost data, even if CA is expressly advised of such loss or damage.

The use of any product referenced in this documentation and this documentation is governed by the end user's applicable license agreement.

The manufacturer of this documentation is Computer Associates International, Inc.

Provided with "Restricted Rights" as set forth in 48 C.F.R. Section 12.212, 48 C.F.R. Sections 52.227-19(c)(1) and (2) or DFARS Section 252.227-7013(c)(1)(ii) or applicable successor provisions.

© 2002 Computer Associates International, Inc.

All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

# Table of Contents


**About This Manual ..... xxxiii**

Who Should Use This Manual..... xxxiii

Notational Conventions ..... xxxiv
- Commands..... xxxiv
- User Entries..... xxxiv
- Icons ..... xxxiv

References and Related Documentation..... xxxv

**Chapter 1 Introduction to Management Services ..... 1-1**

What is Management Services?..... 1-1
- Operator Console ..... 1-2
- Security ..... 1-2
- Print Management ..... 1-2
- Communications ..... 1-3
- Alert Monitoring ..... 1-3
- Broadcasting..... 1-3
- Report Writing ..... 1-3
- Application Development ..... 1-3
  - Network Control Language..... 1-4
  - Managed Object Development Services..... 1-4
- System Support ..... 1-4
- An Overview of the NetMaster, NetSpy, and SOLVE Products ... 1-5
- Management Services Operating Environments and Supported Terminals..... 1-5

## Chapter 2 Customizing Management Services ..... 2-1


About Customizing Management Services ..... 2-2

Using JCL Parameters to Customize Management Services..... 2-2
Displaying and Changing JCL Parameter Settings ..... 2-2

Identifying the System to Users..... 2-3
Identifying Domains and Panels ..... 2-3

Customizing Your Management Services Region Using ICS ..... 2-4
About Initialization and Customization Services..... 2-4
What Are Parameter Groups? ..... 2-4
Logging On for First-time Customization..... 2-5
Dealing with Initialization Failures..... 2-6
Using the Initialization In Progress Dialog..... 2-6
Product-related ICS Parameter Groups ..... 2-7

Customizing OCS Function Key Settings ..... 2-8
Setting Local Function Keys ..... 2-8
Setting Global Function Keys ..... 2-8
Listing Function Key Settings ..... 2-8
Default OCS Function Key Settings ..... 2-9

Updating System Parameters for a Session ..... 2-9
Displaying and Changing System Parameters ..... 2-10
Customizing Management Services Options ..... 2-10
Initialization Options..... 2-11
Restricting SYSPARMS Command Usage..... 2-11
Product-related SYSPARMS Command Operands ..... 2-12

Using Migration ID Exits to Customize Console Management ..... 2-12
Allowing Dynamic Determination of the Need for Migration IDs 2-12

Customizing Command Authority Levels ..... 2-13
Changing Command Authority Levels ..... 2-13
Disabling Commands ..... 2-14
Replacing Commands with NCL Procedures ..... 2-14

Equating Command Strings..... 2-15
Replacing Commands After System Initialization ..... 2-15

Implementing the Timeout Facility ..... 2-15
Two-level Timeouts ..... 2-16
Implementing Timeouts ..... 2-16
Specifying System-level Timeout Intervals ..... 2-16
Procedure—Setting the First-level Timeout ..... 2-16
Procedure—Setting the Second-level Timeout..... 2-17

Specifying System-level Timeout Actions ..... 2-17
Procedure—Setting the First-level Action ..... 2-17
Procedure—Setting the Second-level Action ..... 2-17
Types of Timeout Action ..... 2-17
Repeating Timeout Actions ..... 2-18
Specifying the Timeout Trigger Event ..... 2-18
Starting the Timeout Facility ..... 2-19
Starting the Timeout Facility Automatically ..... 2-19
Changing Timeout Parameters Dynamically ..... 2-19
Suppressing Timeout Activity ..... 2-20
Other Timeout Considerations ..... 2-20

Customizing National Language Character Set Support ..... 2-20
What National Language Support Affects ..... 2-21
Determining a User's Language Code ..... 2-21

Customizing Management Services for Use with DBCS ..... 2-21
Setting up Management Services to Support DBCS ..... 2-22
Identifying DBCS Devices ..... 2-22
Specifying a DBCS Device Dynamically ..... 2-22
IBM and Fujitsu DBCS Implementation ..... 2-23
DBCS Text Strings ..... 2-23

**Chapter 3 Implementing Security ..... 3-1**

About Security ..... 3-2
UAMS ..... 3-2
External Security Exit ..... 3-3
Partial Security Exit ..... 3-3
Supplied Partial Security Exit ..... 3-3
Full Security Exit ..... 3-4
Considerations When Using a Security Exit ..... 3-4

Implementing Security for the First Time ..... 3-5

Defining Users to the System ..... 3-5

Defining a User ID ..... 3-6
Procedure ..... 3-6
Maintaining User IDs ..... 3-7
Procedure ..... 3-7
Forcing a Password Change for a User ID ..... 3-8
Procedure ..... 3-8

Defining a Model User ID ..... 3-9
How It Works ..... 3-9

Defining a System Console User ID ..... 3-10
Defining the System Console User ID in OS/390 or Later Environments ..... 3-10
Default OPER Environment ..... 3-10
Actual Console Environments ..... 3-11
Receiving Command Replies on the System Console ..... 3-13
Defining the System Console User ID in VM Environments ..... 3-13
Using ROF with System Consoles ..... 3-14
MSGPROC and System Console User IDs ..... 3-14
Unsolicited Output to the System Console ..... 3-14

Defining Background Environment User IDs ..... 3-15
Initialization User IDs ..... 3-15
Initialization Privileges ..... 3-16
Limiting Background Environment Authorities ..... 3-16
Using ROF with Background Environments ..... 3-16
MSGPROC and Background Environment User IDs ..... 3-16

Accessing User ID Definitions Using NCL ..... 3-17
Accessing User ID Information ..... 3-17

Providing a Customized Primary Menu ..... 3-18
About the Supplied Primary Menu Procedure ..... 3-18
Expired Password Procedure ..... 3-18
Reconnection Menu ..... 3-19
Displaying the Primary Menu ..... 3-19
Altering the Primary Environment ..... 3-21
Supporting Single Option Users ..... 3-21

Defining Libraries and Paths ..... 3-22
Procedure—Creating a Panel Library ..... 3-22

**Chapter 4 Implementing Additional Security ..... 4-1**

About Implementing Additional Security ..... 4-2

Implementing an External Security Manager for User Definition ..... 4-2
Functions Performed by the Exit ..... 4-2
Controlling Access to Your System ..... 4-3
EIP Logons ..... 4-3
ROF Logons ..... 4-4
Operating System Console Logon ..... 4-4
System Environment Logons ..... 4-4
APPC User Region Logon ..... 4-4
User ID Modeling ..... 4-5

Allowing Logon Verification ..... 4-6
Using Model User IDs ..... 4-6
APPC Link Verification ..... 4-7
Allowing Users to Change Their Password ..... 4-7
Allowing User ID Information to be Retrieved ..... 4-7
Allowing Updates of User IDs ..... 4-8
Adding Additional Security Functions ..... 4-8
Allowing Listing of User IDs—Full Security Exit Only ..... 4-10
Allowing User IDs to Be Added ..... 4-10
Allowing User IDs to Be Deleted ..... 4-10
Accessing User ID Attributes ..... 4-11

Implementing Security for File Access ..... 4-11
Activating the NCL Authorization Exit ..... 4-12
Errors in the Exit ..... 4-12
Parameters Passed to the Exit ..... 4-13
Pre-loading the NCL Authorization Exit ..... 4-13
Providing Additional Checking in the NCL Authorization Exit ..... 4-13
Correlating Authorization with Security Exit Authorization ..... 4-14
Specifying UDB Authorization ..... 4-14
Determining Access Levels from NCL Procedures ..... 4-15
Specifying Authorization to Write SMF Records ..... 4-16
Return Codes from &SMFWRITE Authorization Requests ..... 4-16
Default SMF Authorization ..... 4-16
Designing Your Exit ..... 4-16
Sample Distributed NCL Authorization Exit ..... 4-17

Implementing INMC Link Security ..... 4-17
The Primary Exit ..... 4-18
The Secondary Exit ..... 4-18

Implementing Dataset Allocation Authority ..... 4-19
Sample Distributed Exit ..... 4-19

Implementing Security for Dataset Services ..... 4-20
Function Calls Made to the Exit ..... 4-20
Sample Distributed Exit ..... 4-20

**Chapter 5 Administering Signed-on Users ..... 5-1**

Introduction ..... 5-2

Using the Active User List Facility ..... 5-3
Actions on the User ID List ..... 5-5

**Chapter 6 Communicating Between Domains Using INMC ..... 6-1**

About Communicating Between Domains ..... 6-2
What Is a Domain? ..... 6-2
What Is INMC? ..... 6-2
Simple INMC Links ..... 6-3
Multipath INMC Links ..... 6-3
Access Methods ..... 6-4
Security ..... 6-4
About Traffic Flow ..... 6-4
Traffic Flow on Simple Links ..... 6-4
Traffic Flow on Multipath Links ..... 6-4
Controlling Links ..... 6-5

About INMC Links ..... 6-5
Static INMC Links ..... 6-5
Dynamic INMC Links ..... 6-6
Establishing INMC Links ..... 6-6
Example ..... 6-7
Routing Commands ..... 6-7
Example ..... 6-7
Communication Recovery ..... 6-7
Static Links ..... 6-7
Dynamic Links ..... 6-8

Controlling INMC Links ..... 6-8
Starting an INMC Link ..... 6-8
Stopping an INMC Link ..... 6-9
Resetting an INMC Link ..... 6-9
Displaying INMC Links ..... 6-10
Improving INMC Link Performance ..... 6-11
Increasing Transmission Buffer Size ..... 6-11
Increasing RU Size in Mode Table Definitions ..... 6-12
Specifying a Particular Route for INMC Traffic ..... 6-12
Using Security with INMC Links ..... 6-13
Diagnosing Problems with INMC Connections ..... 6-13

Planning INMC Links ..... 6-13
Planning Simple-mode Links ..... 6-14
Planning Rotate and Backup Mode Multipath Links ..... 6-15
Planning Preferential Mode Multipath Links ..... 6-17
Traffic Synchronization on Rotate and Backup and Preferential Multipath Links ..... 6-18

Maintaining Link Definitions ..... 6-18
Displaying INMC Link Definitions ..... 6-19
Creating an INMC Link Definition ..... 6-20
Creating a Simple-mode Link Definition ..... 6-20
Creating a Rotate and Backup INMC Link Definition ..... 6-21
Creating a Preferential INMC Link Definition ..... 6-22
Modifying an INMC Link Definition ..... 6-23
Deleting an INMC Link Definition ..... 6-23

Providing Centralized Control of Connected Domains ..... 6-24
Using ROF for Centralized Control of Remote Domains ..... 6-24
Defining User IDs for ROF ..... 6-24
Including Simplified Command Definitions in INIT and READY ..... 6-24
Using ROF with NCL Processes ..... 6-25
Using ROF from Background Environments ..... 6-25
Using ROF from the System Console ..... 6-25
Using ISR for Centralized Control of Remote Domains ..... 6-25
Planning for ISR ..... 6-26
Establishing ISR Conversations ..... 6-26
Monitoring the Status of ISR ..... 6-27
Controlling the Type of Data Received Through ISR ..... 6-28
Running ISR Data Flow Without a Procedure ..... 6-29

**Chapter 7 Using APPC to Communicate with Other Systems ..... 7-1**

About APPC ..... 7-2
Transaction Programs and Conversations ..... 7-2
Conversations and Sessions ..... 7-3
Concurrent Conversations ..... 7-3
Sharing a Single Session ..... 7-3
Conversation Handling ..... 7-3
Session Polarity ..... 7-3
Conversation Types ..... 7-4
APPC Option Sets ..... 7-4
Management Services APPC Option Set Support ..... 7-4
Supported APPC Products ..... 7-6

Defining APPC Links ..... 7-6
Defining Dynamic APPC Links ..... 7-6
APPC Table Requirements ..... 7-7
How APPC Control Tables Interact ..... 7-8

Defining APPC Tables to Management Services ..... 7-9
Defining a Transaction Control Table (TCT) ..... 7-9
Defining Transactions ..... 7-10
Defining a Qualified Transaction ..... 7-10
Generic Transactions ..... 7-10
Specifying a Default Destination ..... 7-11
Maintaining TCT Definitions ..... 7-11
Defining a Dynamic Link Table (DLT) ..... 7-12
Specifying an Option Set ..... 7-12
Maintaining DLT Definitions ..... 7-12
Defining an Option Set Control Table (OSCT) ..... 7-13
Nominating Mode Names ..... 7-13
Maintaining OSCT Definitions ..... 7-13
Defining a Mode Control Table (MCT) ..... 7-13
VTAM Logmodes ..... 7-14
Specifying Modes for Single Session Links ..... 7-14
Specifying Modes for Parallel Session Links ..... 7-15
Default Modes ..... 7-15
Maintaining MCT Definitions ..... 7-15

Implementing Security for APPC Links ..... 7-16
Link Level Security ..... 7-16
Conversation Level Security ..... 7-16
Running a Secured Transaction ..... 7-17
Security Options ..... 7-17
Conversation Level Security and NCL Procedure Environments.. 7-18
Specifying Security for APPC Links with Remote Systems ..... 7-18
APPC Region Use ..... 7-19

Activating APPC Links ..... 7-19
Starting Links Manually ..... 7-19
Deactivating Links Manually ..... 7-19
Starting Links Automatically ..... 7-20
Session Initiation Requests ..... 7-20
How Session Establishment Works ..... 7-22
Session Selection for Conversations ..... 7-23
Deactivating Links Automatically ..... 7-24

APPC Link Definition Examples ..... 7-24
Example 1—Running Conversations Within the Same Domain ... 7-25
Example 2—Running a Link Between Two Domains with Already Verified Security ..... 7-25
Example 3—Running an APPC Link Between a Domain and a Remote LU ..... 7-26

Example 4—Running a Qualified Transaction ..... 7-27
Example 5—Running a Generic Transaction ..... 7-28

APPC Application Programming Facilities..... 7-29

**Chapter 8 Partitioning VTAM Networks ..... 8-1**

About the Network Partitioning Facility (NPF) ..... 8-2
About Command Partitioning ..... 8-2
About Message Partitioning..... 8-3
Resource Table Status..... 8-3
Assigning to Multiple Operators..... 8-3

Storing Network Partitioning Information..... 8-4
Storing Network Partitioning Information in OS/VS Systems ..... 8-4
Storing Network Partitioning Information in VM Systems ..... 8-4

VTAM Message Table Requirements ..... 8-5
VTAM Message Tables in OS/VS Systems..... 8-5
VTAM Message Tables in VM Systems ..... 8-5

Defining NPF Control Members ..... 8-6
Combined Control Members..... 8-6
Coding Control Members..... 8-6
Syntax Rules for Control Member Statements ..... 8-7
Syntax Errors ..... 8-8

Defining NPF Resource Tables ..... 8-8
Coding Resource Tables ..... 8-9
Syntax Rules for Coding Resource Tables ..... 8-10
Loading Tables..... 8-11
Testing New Tables..... 8-12
Dealing with Table Errors..... 8-12
Replacing Tables Online..... 8-12
Deleting Tables ..... 8-12
Displaying Tables..... 8-13
Changing the Status of a Message Table ..... 8-13

Defining Network Partitioning for the System Console..... 8-14

Using NPF when Logging on to Remote Systems ..... 8-14

**Chapter 9 Customizing MAI-OC ..... 9-1**

About MAI-OC ..... 9-2
Using MAI-OC ..... 9-2

Using System Parameters to Customize MAI-OC ..... 9-3

Using MAI-OC with Multiple Domains ..... 9-4

Using an MAI Installation Exit ..... 9-4

Defining MAI-OC to Target Applications ..... 9-4

Sample LU Definitions ..... 9-5
Defining MAI-OC to a CICS/VS System ..... 9-5
Definition MAI-OC to an IMS/VS System ..... 9-6
Defining MAI-OC to a JES2 System ..... 9-6
Defining MAI-OC to a JES3 System ..... 9-7

**Chapter 10 Customizing Broadcast Capabilities ..... 10-1**

About Customizing Broadcast Capabilities ..... 10-2

Listing System Group Definitions ..... 10-2
Adding a System Group Definition ..... 10-3

**Chapter 11 Customizing Printer Capabilities ..... 11-1**

About Customizing PSM ..... 11-2

Accessing PSM ..... 11-3

Assigning Printer Definitions ..... 11-3
Adding Printer Definitions ..... 11-4
Listing Printer Definitions ..... 11-5

Specifying Paper Definitions ..... 11-6
Adding Paper Definitions ..... 11-7
Listing Paper Definitions ..... 11-8

Specifying Control Characters ..... 11-9
Displaying the Setup Definition Menu ..... 11-9
Adding Control Characters ..... 11-10
Listing Control Characters ..... 11-11

Specifying Default Printers ..... 11-13
Displaying the Default Printer Assignment Menu ..... 11-13
Adding a Default Printer for a User ID ..... 11-14
Listing Default Printers ..... 11-14

Maintaining Your Printers ..... 11-16
Accessing the Administrative Functions Menu ..... 11-16
Maintaining Printer Defaults ..... 11-16
Updating the Defaults Definition ..... 11-17
Clearing the Printer Spool ..... 11-18
Date Formats ..... 11-19
Distributed Banner Exit Procedure ..... 11-20

Sending Print Requests to NET/MAIL ..... 11-20
Prerequisites for Using the Printer Exit Procedure ..... 11-20
Who Does the Mail Come From? ..... 11-21
Output Segmentation ..... 11-21
Using the Remote Access Facility ..... 11-21
Output Formatting Considerations ..... 11-22
$PSML78X and $PSML79X Parameters ..... 11-22
Sample Printer Exit Definitions ..... 11-26
Sample 1 ..... 11-26
Sample 2 ..... 11-26
Sample 3 ..... 11-27

Sending Print Requests to a Dataset ..... 11-28
How the Procedures Process a Print Request ..... 11-28
$PSDS81X and $PSDS81Z Parameters ..... 11-28
Sample Printer Exit Definition ..... 11-29

**Chapter 12 Implementing Activity Logs ..... 12-1**

About the Activity Logs ..... 12-2

About the Online Activity Log ..... 12-3

Implementing Online Activity Logging ..... 12-3
Using Additional Log Files ..... 12-3

Browsing Online Logs ..... 12-5

Administering Online Activity Log Files ..... 12-6

Swapping the Online Log ..... 12-6

Dealing with Errors on the Online Log ..... 12-7

Recording Additional Information on the Online Log ..... 12-7

Using a Log Exit for the Online Log ..... 12-8
Variables Available to the Activity Log Exit ..... 12-8
Enabling the Log Exit ..... 12-9

Replacing Your Online Logging Procedure ..... 12-9
Writing Your Own Log Browsing Procedure ..... 12-10
Writing Your Own Logging and Browsing Procedures..... 12-11
Implementing Your Own Logging and Browsing Procedures..... 12-11

About the Hardcopy Activity Log ..... 12-11
The Activity Log in OS/VS Systems ..... 12-11
The Activity Log in VM/GCS Systems ..... 12-11
Sending Log Information to the Spooled Printer..... 12-12
Sending Log Information to a CMS File ..... 12-12
Format of Logged Information..... 12-12
Format of Logged Timer Initiated Commands ..... 12-13
Format of Logged Commands Executed in Background Environments ..... 12-13
Format of Logged Commands from NCL Procedure Dependent Environment..... 12-13
Format of the Hardcopy Log..... 12-13

Swapping the Hardcopy Log ..... 12-14
Swapping the Hardcopy Log in OS/VS Systems..... 12-14
Swapping the Hardcopy Log in VM/GCS Systems..... 12-14

Wrapping the Hardcopy Log Datasets (OS/VS)..... 12-15

Cross Referencing Hardcopy Logs..... 12-15

Dealing with I/O Errors on the Hardcopy Log (OS/VS) ..... 12-16

Recording Additional Information on the Hardcopy Log ..... 12-16

Writing to the System Log (OS/VS)..... 12-16

**Chapter 13 Implementing the SOLVE Subsystem Interface (SOLVE SSI) ..... 13-1**

About SOLVE SSI..... 13-2

Implementing SOLVE SSI ..... 13-2
Checking the JCL, Datasets, and Parameters..... 13-3
Checking the JCL..... 13-3
Checking Dataset Allocations..... 13-3
Checking Initialization Parameters ..... 13-5

Activating SOLVE SSI..... 13-9
Connecting to SOLVE SSI..... 13-9
Specifying the Retry Interval..... 13-10

Terminating SOLVE SSI..... 13-10

Executing SOLVE SSI Commands ..... 13-11
SSI Command Descriptions ..... 13-11

Using PPI with SOLVE SSI ..... 13-12
Implementing PPI..... 13-12
PPI-specific Parameter Descriptions..... 13-13
Activating PPI ..... 13-14
PPI Command Descriptions ..... 13-15

**Chapter 14 Recording CPU Usage ..... 14-1**

About NCL CPU-time Accounting ..... 14-2

Implementing NCL CPU-time Accounting ..... 14-2
Generation of SMF Records..... 14-2
Adjusting Threshold Levels and Report Intervals ..... 14-3
SMF Record Format..... 14-3

Starting and Stopping CPU-time Accounting..... 14-5
The USERACCT Command ..... 14-5

Displaying the Status of NCL CPU-time Accounting ..... 14-7

**Chapter 15 Implementing User Programs ..... 15-1**

About the SUBSYS Facility ..... 15-2
Using &CALL Without SUBSYS ..... 15-2
Using &CALL With SUBSYS..... 15-2
Extensions to &CALL..... 15-3
Uses of SUBSYS..... 15-3

Sending Parameter Lists to the Subsystem ..... 15-3

Controlling Subsystems ..... 15-4
Defining a Subsystem ..... 15-4
Starting a Subsystem ..... 15-4
Stopping a Subsystem ..... 15-5
Forcing a Subsystem to Stop..... 15-5
Deleting a Subsystem ..... 15-5
Reloading the Program..... 15-5
Displaying the Status of a Subsystem ..... 15-6

Writing a SUBSYS Program ..... 15-6
Subsystem Program Considerations..... 15-6
Writing a Subsystem Program in High-level Languages..... 15-8

**Chapter 16 Tuning System Performance..... 16-1**

About Enhancing Performance..... 16-2

Tuning at the System Level..... 16-2

Distributing Management Services Functions Across Different Domains..... 16-3
Using JCL Parameters to Include and Exclude Functions..... 16-3
Extending Your JCL Parameter Capacity..... 16-4

Tuning Your VTAM Interface..... 16-4

Tuning Panel Use..... 16-4
Tuning Access to Panels..... 16-5
Tuning Storage Limits for Panel Sends..... 16-5
Monitoring Panel Send Storage Limits..... 16-6
Reducing the Amount of Storage Needed When Using EASINET..... 16-6

Improving NCL Procedure Usage..... 16-7
Monitoring NCL Procedure Loading Activity..... 16-7
Sharing NCL Procedures Between Users..... 16-7
Specifying a Retention Storage Limit..... 16-8
Preloading NCL Procedures..... 16-8
Other NCL Procedure Considerations for Increasing Performance..... 16-8

Tuning Your VSAM Datasets..... 16-9
Using Buffer Sharing..... 16-9
Using Deferred Write Capabilities..... 16-10
Performing VSAM I/O in a Subtask..... 16-10
Tuning Record Sizes..... 16-11
Tuning Database Activity..... 16-11
Monitoring VSAM Tuning..... 16-12
Monitoring LSR Tuning..... 16-12
Monitoring Dataset Activity Tuning..... 16-13

Tuning Communication Between Domains..... 16-14
Tuning INMC..... 16-14
Tuning ROF..... 16-15
Tuning FTS..... 16-15
Tuning ISR..... 16-15

Controlling Message Flow in OCS..... 16-16

Performance Considerations When Writing NCL..... 16-16
Using Commands to Control Resource Consumption by NCL..... 16-17

## Chapter 17 Using the SYSCMD Facility ..... 17-1

Using SYSCMD in an OS/390, z/OS, MSP, or VOS3 Environment ... 17-2
Setting Console Timeouts ..... 17-2
Locking Consoles to a Specific Environment ..... 17-2
Clearing a Lock ..... 17-2
Simulating Command Issue from a Specific Console ..... 17-3

Using SYSCMD in an &INTCMD Environment ..... 17-4
Clearing a Lock in an &INTCMD Environment ..... 17-4

Prefixing Messages ..... 17-4

Facilitating User-Generated Messages ..... 17-5
Insertion Points ..... 17-5
Using the &WTO NCL Statement ..... 17-5
Recommendations for Using &WTO ..... 17-5
Suggested Uses for &WTO ..... 17-6
Using the &WTOR NCL Statement ..... 17-6
Recommendations for Using &WTOR ..... 17-6
Suggested Use for &WTOR ..... 17-7
Using the &DOM NCL Statement ..... 17-7
Recommendations for Using &DOM ..... 17-7
Suggested Use for &DOM ..... 17-7

SYSCMD Command Summary ..... 17-8

SYSCMD NCL Verb and System Variable Summary ..... 17-9

## Chapter 18 Implementing the Data Space Manager ..... 18-1

About the Data Space ..... 18-2
Areas of the Data Space ..... 18-3

About the DIV Dataset ..... 18-4

About the Data Space Manager ..... 18-5

Implementing the Data Space ..... 18-5
Initialization Parameters ..... 18-5

Using the Data Space Manager ..... 18-7
Starting the Data Space Manager ..... 18-7
Stopping the Data Space Manager ..... 18-7
Displaying Data Space Manager Status and Statistics ..... 18-8
Producing Dumps as Diagnostic Tools ..... 18-8
Formatted Dumps ..... 18-9
Unformatted Dumps ..... 18-9

**Chapter 19 Customizing and Using the NetMaster-to-NetSpy Interface ..... 19-1**

Managing NetMaster-to-NetSpy Connections ..... 19-2

Managing NetSpy Alerts and Monitors ..... 19-3
Defining NetSpy User Alert Monitors ..... 19-3
Managing NetSpy User Alert Monitors in NetMaster ..... 19-4

Issuing NetSpy Commands ..... 19-5

Using ICS Parameters to Customize the NetMaster-to-NetSpy Interface ..... 19-7

**Appendix A Connecting Terminals to Management Services ..... A-1**

Supported Terminals ..... A-2

How Management Services Accepts Terminal Connections ..... A-3

Extended Attributes Support ..... A-4

Specifying Screen Sizes ..... A-4

Connecting Hitachi 560/20 Devices ..... A-5

Specifying Logmode Table Definitions ..... A-6

**Appendix B JCL Parameters ..... B-1**

JCL Parameter Summary Table ..... B-1

JCL Parameter Descriptions ..... B-3

**Appendix C The UTIL0028 Utility ..... C-1**

About UTIL0028 ..... C-1

Using UTIL0028 ..... C-2
Specifying the U28SYSIN Dataset ..... C-2

UTIL0028 Commands ..... C-3
General Commands ..... C-3
Dynamically Allocating a Dataset ..... C-4
Dynamically Allocating a SYSOUT Dataset ..... C-5

UTIL0028 Processing ..... C-7

**Appendix D SYSPARMS Operands ..... D-1**

Generic SYSPARMS Summary Table ..... D-2

Generic SYSPARMS Operand Descriptions ..... D-12

SYSPARMS Operands for NEWS ..... D-53

SYSPARMS Operands for NTS ..... D-54

SYSPARMS Operands for SYSCMD ..... D-58

**Appendix E UAMS Panel Descriptions ..... E-1**

The User Details Panel ..... E-1

The User Authorities Panel ..... E-4

The Access Authorities Panel ..... E-7

The MODS Details Panel ..... E-10

The Installation Attributes Panel ..... E-11

The OCS Details Panel ..... E-11

The Notification Details Panel ..... E-14

The Network Management Details Panel ..... E-16

Print Services Manager Details Panel ..... E-17

Report Writer Details Panel ..... E-19

**Appendix F User ID Security Exit Support ..... F-1**

External Security Packages ..... F-2

Writing Your Own User ID Security Exit ..... F-3
- Exit Execution ..... F-5
- Supported Exit Calls ..... F-6

The System Initialization Parameter List ..... F-7
- Return Codes from Initialization Call ..... F-8

The System Close Down Parameter List ..... F-9
- Return Codes from Closedown Call ..... F-10

The Logon Request Parameter List ..... F-10
- Return Codes from Logon Call ..... F-14

The Logoff Request Parameter List ..... F-16
- Return Codes from Logoff Calls ..... F-17

The Logon Verification Call Parameter List ..... F-18
- Return Codes from Logon Verification Call ..... F-20

The Change Password Parameter List ..... F-22
Return Codes from Change Call ..... F-24

The Return User ID Information Parameter List ..... F-25
Return Codes from Return User ID Information Call ..... F-26

The Update User ID Parameter List ..... F-27
Return Codes from Update User ID Information Call ..... F-29

The &SECCALL EXIT Parameter List ..... F-29
Return Codes from &SECCALL EXIT Call ..... F-31

The Return Sequential User ID Information Parameter List ..... F-32
Return Codes from Return Next User ID Information Call ..... F-33

The Add User ID Parameter List ..... F-34
Return Codes from the Add User ID Call ..... F-36

Delete User ID Parameter List ..... F-37
Return Codes from the Delete User ID Call ..... F-38

**Appendix G Structured Fields ..... G-1**

About Structured Fields ..... G-1

Format of Structured Fields ..... G-2
Example ..... G-2
Updating a Structured Field ..... G-3
Structured Field Error Conditions ..... G-4
Structured Field Sequences ..... G-4

Structured Field Descriptions ..... G-4

Structured Field Key X‘0010’—User ID Name ..... G-5

Structured Field Key X‘0011’—User Name ..... G-5

Structured Field Key X‘0012’—User Location ..... G-5

Structured Field Key X‘0013’—User Telephone Number ..... G-6

Structured Field Key X‘0014’—User Language Code ..... G-6

Structured Field Key X‘0015’—User ID Suspend Date ..... G-6

Structured Field Key X‘0016’—Terminal Restrictions ..... G-7

Structured Field Key X‘0017’—Timeout Control ..... G-7

Structured Field Key X‘0018’—Date/Time User ID Last Updated ..... G-8

Structured Field Key X‘0019’—Multiple Signon Capability ..... G-8

Structured Field Key X‘001A’ —Group Definition for User ..... G-9

Structured Field Key X‘001B’—User ID Definition Type ..... G-9

Structured Field Key X‘001C’ —User Password Expiry Action Indicator ..... G-10
Structured Field Key X‘0020’ —OCS Access Privilege ..... G-10
Structured Field Key X‘0021’—Broadcast Services Privilege ..... G-10
Structured Field Key X‘0022’ —Network Services Access Privilege. G-11
Structured Field Key X‘0023’—System Support Privilege ..... G-11
Structured Field Key X‘0024’—CAS Panel Maintenance Privilege ... G-12
Structured Field Key X‘0025’—FTS Access Privilege ..... G-12
Structured Field Key X‘0026’—NEWS Access Privilege ..... G-12
Structured Field Key X‘0027’—MAI-FS Access Privilege ..... G-13
Structured Field Key X‘0028’—User Services Procedure Name ..... G-13
Structured Field Key X‘0029’—User’s NCL Procedure Library ..... G-14
Structured Field Key X‘002A’—UAMS Access Privilege ..... G-14
Structured Field Key X‘002B’—Operations Management Privilege... G-14
Structured Field Key X‘002C’—TSO Autologon Privilege ..... G-15
Structured Field Key X‘002D’—NCS Access Privilege ..... G-15
Structured Field Key X‘002E’ —User’s SPLIT/SWAP Privilege ..... G-15
Structured Field Key X‘002F’—Library Services Path Name ..... G-16
Structured Field Key X‘0030’—User’s Time Zone Name ..... G-16
Structured Field Key X‘0050’—OCS Command Authority Level ..... G-16
Structured Field Key X‘0051’—OCS Monitor Status ..... G-17
Structured Field Key X‘0052’—NPF Command Member ..... G-17
Structured Field Key X‘0053’—MSGPROC Member ..... G-17
Structured Field Key X‘0054’—OCS Mode Initial Command ..... G-18
Structured Field Key X‘0055’—PPO Message Receipt Option ..... G-18
Structured Field Key X‘0056’—PPO Severity Level ..... G-18
Structured Field Key X‘0057’—NPF Message Restriction Option ..... G-19
Structured Field Key X‘0058’—Message Code Value ..... G-19
Structured Field Key X‘0059’—OCS MSG Message Receipt ..... G-20
Structured Field Key X‘005A’—OCS Unsolicited Message Receipt Option ..... G-20
Structured Field Key X‘005B’—Resource List Member ..... G-20
Structured Field Key X‘005C’—User Timeout (1) Period ..... G-21

Structured Field Key X‘005D’—User Timeout (2) Period ..... G-21
Structured Field Key X‘005E’—User Timeout (1) Action ..... G-21
Structured Field Key X‘005F’—User Timeout (2) Action ..... G-22
Structured Field Key X‘0060’—User’s APPC Access Key..... G-22
Structured Field Key X‘0061’—User’s APPC Access Lock ..... G-23
Structured Field Key X‘0070’ —Installation Attribute Field 1..... G-23
Structured Field Key X‘0071’—Installation Attribute Field 2..... G-23
Structured Field Key X‘0072’—Installation Attribute Field 3..... G-24
Structured Field Key X‘0073’—Installation Attribute Field 4..... G-24
Structured Field Key X‘0074’—Installation Attribute Field 5..... G-24
Structured Field Key X‘0075’—Installation Attribute Field 6..... G-25
Structured Field Key X‘0076’—Installation Attribute Field 7..... G-25
Structured Field Key X‘0077’—Installation Attribute Field 8..... G-25
Structured Field Key X‘0078’—Installation Attribute Field 9..... G-26
Structured Field Key X‘0079’—Installation Attribute Field 10..... G-26
Structured Field Key X‘0080’—User’s Access to INFO/MASTER Maintenance Functions ..... G-26
Structured Field Key X‘0081’—User’s Access to Information Services ..... G-27
Structured Field Key X‘0090’—User’s Access to NCPView ..... G-27
Structured Field X‘0150’—Define NEWS Reset Privilege ..... G-28
Structured Field X‘0151’—Define NTS Access Privilege..... G-28
Structured Field Key X‘0180’—User’s AOM Message Delivery and Routing Codes ..... G-29
Structured Field Key X‘0181’—User’s AOM MVS SYSCMD Console Authority ..... G-30
Structured Field Key X‘0182’ —User’s AOM MSG Level..... G-31
Structured Field Key X‘0183’—User’s AOM VM SYSCMD Authority ..... G-32
Structured Field Key X‘0185’ —User’s AOM VOS3/JSS4 SYSCMD Command Authority ..... G-32
Structured Field Key X‘0200’—User’s MAI-FS Privilege Class ..... G-33
Structured Field Key X‘0201’—User’s MAI-FS Model User ID ..... G-33
Structured Field Key X‘0202’—User’s MAI-FS A and E Command Capability ..... G-33

Structured Field Key X‘0500’—User’s PSM Primary Menu Access .. G-34

Structured Field Key X‘0501’—User’s PSM Maintenance Access..... G-34

Structured Field Key X‘0502’—User’s PSM Administration Access . G-34

Structured Field Key X‘0503’—User’s PSM Ability to Change Print Request Priority ..... G-35

Structured Field Key X‘0504’—User’s PSM Queue Access for All Print Output ..... G-35

Structured Field Key X‘0505’—User’s PSM Queue Access for Their Own Print Output..... G-36

Structured Field Key X‘0510’ —User’s Panel Command Access Authority..... G-36

Structured Field Key X‘0511’ —User’s Management Services Access ..... G-37

Structured Field Key X‘0520’—User’s Notification Details (First Rule)..... G-38

Structured Field Key X‘0521’—User’s Notification Details (Second Rule) ..... G-39

Structured Field Key X‘0522’—User’s Notification Details (Third Rule) ..... G-40

Structured Field Key X‘0523’—User’s Notification Details (Fourth Rule) ..... G-41

Structured Field Key X‘0530’—TCP/IP Services Access Privilege.... G-42

Structured Field Key X‘0550’—User’s Report Writer Primary Menu Access ..... G-42

Structured Field Key X‘0551’—User’s Report Writer Administration Access ..... G-43

Structured Field Key X‘0552’—User’s Report Writer Maintenance Access ..... G-43

Structured Field Key X‘0553’—User’s Report Writer Public Report Access ..... G-43

Structured Field Key X‘0554’—User’s Report Writer Access to Their Own Reports ..... G-44

Structured Field Key X‘0555’—User’s Report Writer Private Report Access for All Users ..... G-44

Structured Field Key X‘0556’—User’s Report Writer Schedule Maintenance Access ..... G-45

Structured Field Key X‘0580’—User’s Access to SOLVE:NetMail ... G-45

Structured Field Key X‘0600’—User’s Access to SOLVE:Central

Applications ..... G-45

Structured Field Key X‘0601’—User’s Access to Managed Objects Development Services (MODS) ..... G-46

Structured Field Key X‘0602’—User’s Access to the MODS Application Register ..... G-46

Structured Field Key X‘0603’—User’s Access to the MODS Administration Function ..... G-46

Structured Field Key X‘0604’—User’s SOLVE:Central Primary Menu ..... G-47

Structured Field Key X‘0605’—User’s Object Services Access ..... G-47

Structured Field Key X‘0606’—User’s Object Services Object Class Specification Access ..... G-47

Structured Field Key X‘0607’—User’s Access to Mapping Services . G-48

Structured Field Key X‘0608’—User’s SOLVE:Central Security Profile ..... G-48

Structured Field Key X‘0609’—User’s Object Services Security Access ..... G-48

Structured Field Key X‘0610’—User’s CAS Menu Maintenance Access ..... G-49

Structured Field Key X‘0611’—User’s CAS List Maintenance Access ..... G-49

Structured Field Key X‘0612’—User’s CAS Panel Domain Maintenance Access ..... G-49

Structured Field Key X‘0613’—User’s CAS Help Maintenance Access ..... G-50

Structured Field Key X‘0614’—User’s CAS Message Maintenance Access ..... G-50

Structured Field Key X‘0615’—User’s CAS Table Maintenance Access ..... G-50

Structured Field Key X‘0616’—User’s CAS Criteria Maintenance Access ..... G-51

Structured Field Key X‘0617’—User’s CAS Command Maintenance Access ..... G-51

Structured Field Key X‘0620’—User’s SOLVE:Problem Security Profile ..... G-51

Structured Field Key X‘0621’—User’s SOLVE:Problem Primary Menu ID ..... G-52

Structured Field Key X‘0640’—User’s SOLVE:Change Security Profile ..... G-52

Structured Field Key X‘0641’—User’s SOLVE:Change Primary Menu ID ..... G-52

Structured Field Key X‘0660’—User’s SOLVE:Asset Security Profile ..... G-53

Structured Field Key X‘0661’ —User’s SOLVE:Asset Primary Menu ID ..... G-53

Structured Field Key X‘0680’—User’s SOLVE:Configuration Security Profile ..... G-53

Structured Field Key X‘0681’—User’s SOLVE:Configuration Primary Menu ID ..... G-54

Structured Field Key X‘0700’—User’s SOLVE:Problem Helpdesk Security Profile ..... G-54

Structured Field Key X‘0701’—User’s SOLVE:Problem Helpdesk Primary Menu ID ..... G-55

**Appendix H Command Authority Levels ..... H-1**

About Command Authority Levels ..... H-1

Command Authority Summary Table ..... H-2

**Appendix I Dataset Authorization Exits Support ..... I-1**

Writing a Dataset Access Authorization Exit ..... I-1
- Registers on Entry to the Exit ..... I-1
- Parameters Passed to NMDSNCHK ..... I-2
  - Identifying the Type of Call ..... I-2
- Return Codes From the Exit ..... I-2

Installing the Dataset Access Authorization Exit ..... I-3
- Installing the Exit in OS/390, z/OS, and MSP Systems ..... I-3
- Installing the Exit in VOS3 Systems ..... I-3
- Installing the Exit in VM Systems ..... I-4

Writing a Dataset Services Authorization Exit ..... I-4
- Exit Environment ..... I-4
- Registers on Entry to the Exit ..... I-5
  - Exit Initialization Call ..... I-5
  - Logical Function Call ..... I-6
  - Exit Termination Call ..... I-7

Installing the Dataset Services Authorization Exit ..... I-7

**Appendix J INMC Security Exit Support..... J-1**

Writing an INMC Security Exit..... J-1
Identifying the Primary Exit to Management Services..... J-2
Identifying the Secondary Exit to Management Services..... J-2
Changing Exit Names Dynamically..... J-2
Registers on Entering INMC Exits ..... J-3

Writing a Primary Exit..... J-3
Specifying Initialization Processing..... J-4
Initialization Call Parameter List..... J-4
Return Codes from Initialization Call..... J-6
Specifying Message Delivery Processing ..... J-8
Delivery Call Parameter List ..... J-8
Return Codes from Delivery Call ..... J-10
Specifying Termination of Link Notification Processing ..... J-11
Notify Call Parameter List..... J-11
Return Codes from Notify Call..... J-13

Writing a Secondary Exit..... J-14
Specifying Initialization Processing..... J-14
Initialization Call Parameter List..... J-14
Return Codes from the Initialize Call ..... J-15
Specifying Message Delivery Processing ..... J-16
Deliver Call Parameter List ..... J-16
Return Codes from Secondary Exit Deliver Call..... J-18
Specifying Termination Processing ..... J-18
Disconnect Call Parameter List ..... J-19
Return Codes from the Disconnect Call ..... J-20

**Appendix K The NCL Authorization Exit Communication Area ..... K-1**

Communication Area ..... K-1

**Appendix L MAI Installation Exit MAIEX02 ..... L-1**

About the MAI Installation Exit ..... L-1

Implementing the MAI Exit ..... L-2
- Reentrant Code ..... L-2
- Storage Subpools ..... L-2
- Exit Serialization ..... L-3
  - First Level Serialization ..... L-3
  - Second Level Serialization ..... L-3
- The Sample Exit ..... L-3

Starting the Exit ..... L-4

Registers on Entry to the Exit ..... L-4

Exit Correlators ..... L-5
- System Correlator ..... L-5
- User Correlator ..... L-5
- Session Correlator ..... L-5
- Security Exit Correlator ..... L-5
- Security Exit User Token ..... L-5

Contents of the Communications Area ..... L-6
- System Initialization Call ..... L-6
- System Closedown Call ..... L-6
- MAI Session Start Call ..... L-7
- ACB Open Call ..... L-8
- MAI Session End Call ..... L-8

Return Codes from the Exit ..... L-9
- System Initialization ..... L-9
- Session Start ..... L-9

**Appendix M Delivery of Unsolicited VTAM Messages ..... M-1**

Severity Classification of VTAM Messages ..... M-1

Delivery Logic of VTAM Messages ..... M-2

**Appendix N Performance and Tuning Commands ..... N-1**

Using Performance and Tuning Commands ..... N-1

SHOW SYSPGT ..... N-2
- Function: ..... N-2
- Use: ..... N-2
- Examples: ..... N-2
- Notes: ..... N-3
- See Also: ..... N-3

SHOW SYSRCT ..... N-4
- Function: ..... N-4
- Use: ..... N-4
- Operands: ..... N-4
- Examples: ..... N-4
- Notes: ..... N-5
- See Also: ..... N-6

SHOW SYSWAIT ..... N-7
- Function: ..... N-7
- Use: ..... N-7
- Examples: ..... N-7
- Notes: ..... N-8
- See Also: ..... N-8

SYSPGT ..... N-9
- Function: ..... N-9
- Use: ..... N-9
- Operands: ..... N-9
- Examples: ..... N-9
- Notes: ..... N-9
- See Also: ..... N-10

SYSRCT ..... N-11
- Function: ..... N-11
- Use: ..... N-11
- Operands: ..... N-11
- Examples: ..... N-12
- Notes: ..... N-12
- See Also: ..... N-13

**Appendix O Using Management Services with XNF ..... O-1**

About XNF ..... O-2
VTAM Versus XNF Terminology ..... O-2

Connecting Terminals to Management Services ..... O-3
Specifying Screen Sizes for 3270 Terminals ..... O-3
Extended Attribute Support ..... O-3
Changing Terminal Attributes ..... O-4

Shutting Down and Restarting XNF ..... O-4

Customizing Management Services ..... O-4
JCL Parameters ..... O-4
SYSPARMS ..... O-5

Implementing Security in an XNF system ..... O-6

Using INMC Links in an XNF System ..... O-6

Using APPC Links in an XNF System ..... O-7

Issuing Management Services (MS) Commands on an XNF System .. O-7

Using NCL Components in an XNF System ..... O-8

**Appendix P Defining User Migration ID Exits ..... P-1**

User Exits for Migration IDs ..... P-2
Why Define a User Exit? ..... P-2
Sample Exit ..... P-2
Activating the User Exit ..... P-2
User Exit Requirements ..... P-2
Input Parameters for User Exits ..... P-3
Return Codes from a User Migration ID Exit ..... P-5
Exit Tracing ..... P-5

**Glossary**

**Index**

# Figures and Tables


Figure 2-1. Display/Update SYSPARMS Values Panel ..... 2-10
Figure 3-1. UAMS : User Details Panel ..... 3-8
Figure 4-1. Representation of &SECCALL Exit Variables to Security Exit 4-9
Figure 5-1. Management Services : Primary Menu ..... 5-3
Figure 5-2. System Support : User ID List ..... 5-4
Figure 6-1. INMC : Link Status List Panel ..... 6-10
Figure 6-2. INMC : Link Status Display Panel ..... 6-11
Figure 6-3. Rotate Link Configuration ..... 6-16
Figure 6-4. Backup Link Configuration ..... 6-17
Figure 6-5. INMC : Link Definition List Panel ..... 6-19
Figure 6-6. INMC : Link Definition Details Panel ..... 6-20
Figure 6-7. INMC : Rotate and Backup Session Details Panel ..... 6-21
Figure 6-8. INMC : Preferential Session Details Panel ..... 6-22
Figure 6-9. SHOW ISR Command Output ..... 6-27
Figure 7-1. APPC Table Interactions During Link Activation ..... 7-9
Figure 8-1. Schematic of NPF Definition Requirements ..... 8-9
Figure 10-1. Broadcast Services : Group List ..... 10-2
Figure 10-2. Broadcast Services : Group Definition ..... 10-3
Figure 10-3. Broadcast Services : Resource List ..... 10-4
Figure 10-4. Broadcast Services : Resource Definition ..... 10-4
Figure 11-1. Print Services Manager ..... 11-3
Figure 11-2. Printer Definition Menu ..... 11-4
Figure 11-3. Printer Definition Panel ..... 11-5
Figure 11-4. Printer Definition List Panel ..... 11-6
Figure 11-5. Form Definition Menu ..... 11-7
Figure 11-6. Form Definition Panel ..... 11-8

Figure 11-7. Form Definition List Panel 1 ..... 11-9
Figure 11-8. Setup Definition Menu..... 11-10
Figure 11-9. Setup Definition Panel 1 ..... 11-11
Figure 11-10. Setup Definition List Panel 1 ..... 11-12
Figure 11-11. Default Printer Assignment Menu ..... 11-13
Figure 11-12. Default Printer Assignment Panel..... 11-14
Figure 11-13. Default Printer Assignment List Panel 1 ..... 11-15
Figure 11-14. PSM : Administration Menu ..... 11-16
Figure 11-15. PSM : Defaults Definition Panel..... 11-17
Figure 11-16. Clear Spool Panel..... 11-18
Figure 12-1. Logging Facilities ..... 12-2
Figure 12-2. The LOGFILES Parameter Group ..... 12-4
Figure 12-3. Allocating an Activity Log File ..... 12-4
Figure 12-4. The Activity Log : Administration Menu ..... 12-6
Figure 12-5. Activity Log Services : Confirm Swap Log Panel..... 12-7
Figure 12-6. The LOGFILES Parameter Group ..... 12-9
Figure 16-1. SHOW VSAMIO Command Display..... 16-10
Figure 16-2. SHOW VSAM Command Display ..... 16-12
Figure 16-3. SHOW LSR Command Display on a Customized System..... 16-13
Figure 16-4. SHOW NDB=ALL Command Display ..... 16-14
Figure 18-1. How Events and Messages Are Passed to Regions ..... 18-2
Figure 18-2. Data Space Layout ..... 18-3
Figure 19-1. NetSpy SNA Agents : NetSpy Connections Panel ..... 19-2
Figure 19-2. NetSpy SNA Agents : NetSpy Commands Panel ..... 19-3
Figure 19-3. NetSpy SNA Agents : NetSpy Monitors List ..... 19-4
Figure 19-4. NetSpy SNA Agents : NetSpy Commands Panel ..... 19-5
Figure 19-5. NetSpy Agent Services : NetSpy Command Arguments Panel 19-5
Figure 19-6. NetSpy SNA Agents : NetSpy Command Results Panel..... 19-6
Figure 19-7. The NETSPYLINKS Parameter Group ..... 19-7
Figure A-1. Successful Terminal Connection ..... A-3
Figure F-1. Management Services—Full Security Exit Linkage..... F-4
Figure F-2. Management Services—Partial Security Exit Linkage..... F-4
Figure F-3. System Initialization Parameter List..... F-7
Figure F-4. System Close Down Parameter List ..... F-9
Figure F-5. Logon Request Parameter List..... F-10
Figure F-6. User Logoff Parameter List ..... F-16
Figure F-7. Logon Verification Parameter List ..... F-18
Figure F-8. Change Password Parameter List..... F-22
Figure F-9. Return User ID Information Parameter List ..... F-25
Figure F-10. Update User ID Parameter List..... F-27
Figure F-11. &SECCALL EXIT Parameter List ..... F-29
Figure F-12. Return SEQUENTIAL User ID Information List..... F-32
Figure F-13. Add New User ID Parameter List..... F-34

Figure F-14. Delete User ID Parameter List ..... F-37
Figure G-1. Structured Field Example 1 ..... G-3
Figure G-2. Structured Field Example 2 ..... G-3
Figure J-1. Primary Exit Initialization Call Parameter List ..... J-5
Figure J-2. Primary Exit Delivery Call Parameter List ..... J-9
Figure J-3. Primary Exit Notify Call Parameter List ..... J-12
Figure J-4. Secondary Exit Initialization Call Parameter List ..... J-15
Figure J-5. Secondary Exit Deliver Call Parameter List ..... J-17
Figure J-6. Secondary Exit Disconnect Call Parameter List ..... J-19

Table 3-1. Assigning a System Console User ID (SYSCONXU=NO)..... 3-12
Table 3-2. Assigning a System Console User ID (SYSCONXU=YES).... 3-12
Table 3-3. Primary Menu Options ..... 3-20
Table 12-1. Activity Log Exit—Variables ..... 12-8
Table 14-1. Standard SMF Record Format ..... 14-4
Table 18-1. Configuration Record Detail Types ..... 18-4
Table B-1. JCL Parameter Summary Table ..... B-1
Table B-2. Values of INT Parameter ..... B-5
Table B-3. Values of OPT Parameter ..... B-6
Table B-4. Product Name Key Values for PROD= on OS/390 and z/OS .. B-8
Table D-1. SYSPARMS Summary Table ..... D-2
Table E-1. Notification Modes ..... E-14
Table F-1. Supported Exit Calls ..... F-6

# About This Manual

Management Services is the base on which NetMaster, NetSpy, and SOLVE products run. The *Management Services Administrator Guide* is designed to provide information on how to implement and administer the features of Management Services.

## Who Should Use This Manual

This document is intended as a guide for those technical personnel responsible for administering Management Services. It is also designed as an aid when planning the implementation of NetMaster, NetSpy, and SOLVE products, so that maximum benefits can be obtained from the system as quickly as possible.

This guide should be used in conjunction with your *Management Services User's Guide*, which gives details of the general operation of Management Services.

**Note**

If you are intending to use systems in an XNF system, then see Appendix O, *Using Management Services with XNF*. This appendix covers those aspects of Management Services that apply only to XNF, and identifies those topics in the body of the manual that do not apply to XNF.

## Notational Conventions

This section explains the conventions used when referring to various types of commands and user entries:

### Commands

All commands in this manual are shown in uppercase. For example, SHOW USERS.

### User Entries

All examples that can be entered by a user are displayed in bold. For example, Enter **O** at the ===> prompt.

### Icons

Icons are used to draw your attention to important information. The following icons are used in this manual:

**Warning**

This icon indicates a warning to alert you to consequences of certain actions.

**Key Concept**

This icon indicates conceptual information to help you understand the task you are performing.

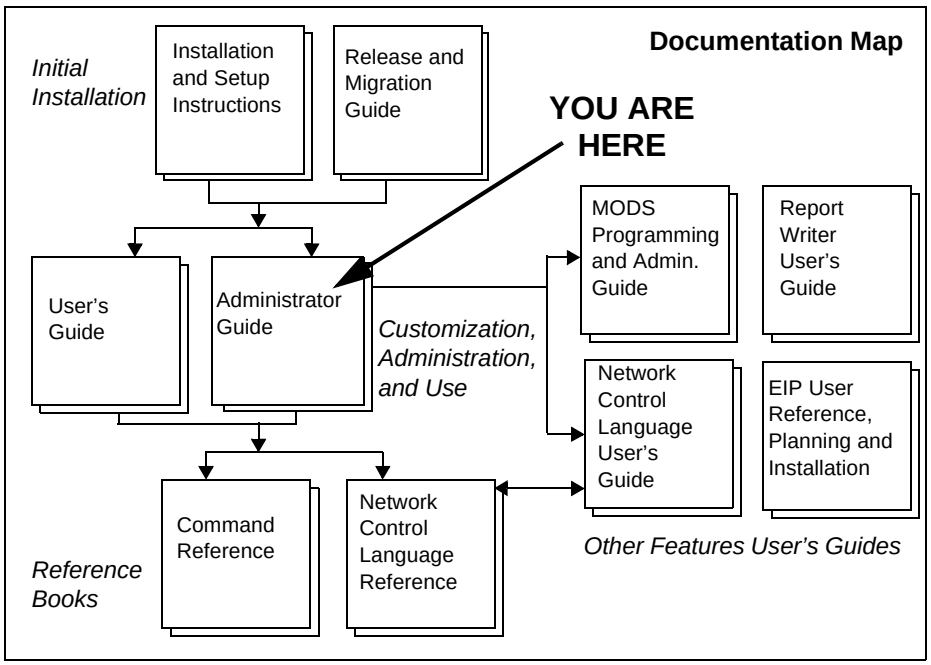
## References and Related Documentation

This manual is part of the Management Services library. The documentation map shows the manuals in the library.

Management Services is installed as part of the installation of your product. See the *Unicenter Mainframe Installation and Setup Instructions (NetMaster and NetSpy)* for details.

**Note**

How you install Management Services depends upon your operating system. For information about installing Management Services on different operating systems and the installation tape format, see the *Unicenter Mainframe Installation and Setup Instructions (NetMaster and NetSpy)* shipped with the installation tape.




Along with the *Management Services Administrator Guide*, the Management Services documentation set is made up of the following manuals:

- *Management Services Version 5.0 Release and Migration Guide*—this guide provides information about the Version 5.0 release of Management Services
- *Management Services User's Guide*—this guide provides information on how to use the features of Management Services
- *Management Services Command Reference*—describes each of the Management Services (MS) commands along with examples of their use

- *Network Control Language Reference*—describes the verbs, built-in functions, and variables that make up the NCL language
- *Network Control Language User's Guide*—describes how to use NCL to write your own code within Management Services
- *MODS Programming and Administration Guide*—describes how to use MODS to create your own panels and applications
- *Report Writer User's Guide*—describes how to use report writer to produce and print your own reports
- *EIP User Reference*—describes how to use the External Interface Package to access Management Services without using a terminal

Some commands and facilities have been documented in this manual, rather than in the *Management Services User's Guide* or the *Management Services Command Reference*.

# 1

# Introduction to Management Services

This chapter introduces Management Services and provides an overview of the services and functions provided as part of Management Services.

## What is Management Services?

The NetMaster, NetSpy, and SOLVE products have been designed with a central core of functions and services. These functions and services are referred to as *Management Services*.

Management Services works in conjunction with any number of enterprise platforms. It provides a range of tools, general services, and facilities and is supplied with each product.

Management Services provides the following services:

- An Operator Console
- Security
- Print Management
- Communications
- Alert Monitoring
- Broadcasting
- Report Writing
- Application Development
- System Support

## Operator Console

Operators Console Services (OCS) provides an operator environment for command entry to allow monitoring and control of your system.

OCS is used in conjunction with other features of Management Services to assist you in this task:

- Activity Log—allows you to access all the commands, messages, or errors that have been issued and logged in the system for any given day
- Network Information Utility File—provides descriptions of errors and codes that are displayed in OCS
- Remote Operator Facility (ROF)—allows you to monitor and control remote systems through OCS
- Network Partitioning Facility (NPF)—allows you to subdivide your network so that different parts are controlled by different operators
- Event Distribution Services (EDS)—allows you to filter out unwanted messages in OCS before they are passed to an application procedure
- Multiple Access Interface-Operator Console (MAI-OC)—allows you to log on to VTAM applications for monitoring and control

For more information see chapters 6, 8, and 12 in this manual, and the *Management Services User's Guide*.

## Security

Security for Management Services is provided by the User ID Access Maintenance Subsystem (UAMS). UAMS provides logon and password checking facilities, and the ability to control the authority and privileges of users.

For more information see Chapter 3, *Implementing Security*, in this manual.

## Print Management

Management Services provides a Print Services Manager (PSM). PSM is a spooling facility which allows you to control the physical printing of the reports your organization generates on JES or network printers.

For more information see Chapter 11, *Customizing Printer Capabilities*, in this manual, and also the *Management Services User's Guide*.

## Communications

Management Services provides several facilities which allow you to communicate between domains and programs, and to collect different types of message flows:

- Inter-Management Services Connection (INMC)—allows you to establish and monitor links between multiple domains
- Advanced Program-to-Program Communication (APPC)—allows you to use the APPC protocol to connect multiple domains
- Inter System Routing (ISR)—allows you to use INMC to provide centralized control at the system level
- Program-to-Program Interface (PPI)—allows you to communicate between programs

For more information on Communication Services, see chapters 6, 7, and 13 in this manual, and also the *Management Services User's Guide*.

## Alert Monitoring

The alert monitor provides an event notification system that tells you that a problem has been detected and that some action needs to be taken.

For more information about using the alert monitor with particular products, see your product manuals.

## Broadcasting

Broadcast Services allows you to send broadcast messages to all users defined to Management Services. Messages can be sent to terminals or can be sent to specific users based on selection criteria.

For more information see Chapter 10, *Customizing Broadcast Capabilities*, in this manual.

## Report Writing

Report Writer provides a facility for defining report layouts and generating reports to suit your particular site requirements.

For more information see the *Report Writers User's Guide*.

## Application Development

Application development services allows you to use facilities provided with Management Services to write your own menus, panels, and applications. Two facilities are provided for this task:

- Network Control Language (NCL)
- Managed Object Development Services (MODS)

### Network Control Language

NCL is the interpretive language which is used within Management Services to develop procedures that can then be executed by Management Services.

NCL also includes the following types of database support:

- User Database support
- NetMaster Database support—by using NDBs
- Sequential database support—by using dataset services
- External database support—by using EDBS

For more information about NCL and its features, see the *Network Control Language User's Guide* and the *Network Control Language Reference*.

### Managed Object Development Services

MODS allows you to create your own applications and develop panels to provide access to them. The following features are available:

- Application Register—the definitions of all applications that are built in MODS must be defined in the application register
- Common Application Services (CAS)—a collection of high-quality, special-purpose NCL routines designed to facilitate program development
- Panel Services—a facility for creating and maintaining full-screen panel definitions
- Object Services—an object-oriented development environment used to define and maintain an application's data and methods
- Mapping Services—a facility that enables programmers to define complex data structures for use by NCL applications
- Administrative Functions—used to maintain MODS control libraries, panel libraries, and object services support functions

For more information about MODS, see the *Managed Object Development Services Programming and Administration Guide*.

## System Support

System support services provides facilities to change system parameters so that you can administer, tune, diagnose, and tailor your domain.

## An Overview of the NetMaster, NetSpy, and SOLVE Products

The NetMaster, NetSpy, and SOLVE products addresses key operational areas of the modern information processing environment. These operational areas (network management, systems administration, and systems operations) require specialized management software to maximize their effectiveness.

- Network Management
  The NetMaster, NetSpy, and SOLVE network management products provide software that optimizes the operation and services of SNA and TCP/IP networks. These products simplify network management tasks and provide user access services.
- Systems Administration
  The SOLVE:Central products help you to manage complex, modern IS environments and provision of services. For example, they allow you manage your financial assets, your system configuration, changes to the system environment, and help desk calls and problems.
- Systems Operations
  The SOLVE:Operations products provide software that manages the automated monitoring of operating system activity and the automation of routine operational tasks.

## Management Services Operating Environments and Supported Terminals

Management Services operates within the following operational environments:

eNetwork Communications Server or ACF/VTAM under:

- OS/390 and z/OS
- VM/ESA

VTAM-G under:

- MSP/EX, MSP/E20AE, and MSP
- FSP/E26 and XSP

ECS/VTAM and XNF under:

- VOS3/AS

Management Services is designed to support large numbers of terminals concurrently. More specifically, it supports 3270-type devices (LU-0 and LU-2), and LU-1 devices. It provides all the necessary services required to establish and disconnect sessions with all 3270 model display terminals (or their equivalent).

On a Hitachi VOS3 system, Management Services also supports connection of 560/20 display terminals.

# 2

# Customizing Management Services

This chapter describes how to customize Management Services by using JCL and system parameters.

**The chapter contains the following topics:**

- About Customizing Management Services
- Using JCL Parameters to Customize Management Services
- Identifying the System to Users
- Customizing Your Management Services Region Using ICS
- Customizing OCS Function Key Settings
- Updating System Parameters for a Session
- Using Migration ID Exits to Customize Console Management
- Customizing Command Authority Levels
- Equating Command Strings
- Implementing the Timeout Facility
- Customizing National Language Character Set Support
- Customizing Management Services for Use with DBCS

## About Customizing Management Services

After you have completed the installation and startup for your product, Management Services is operational at a basic level. To customize Management Services to suit your operational requirements, you need to:

- Set additional JCL parameters
- Include commands in your INIT procedure
- Include SYSPARMS commands in your INIT procedure
- Use ICS parameter groups to customize your region

The changes that you make to your JCL and INIT procedure come into effect when you restart your region.

The following sections describe how to customize the various features of Management Services.

## Using JCL Parameters to Customize Management Services

JCL parameters provide the first point of customization of your region. You use JCL parameters to set information such as the names of your INIT and READY procedures and the types of security exit to be used by the region.

The way that this information is supplied depends on the operating system:

- In OS/VS systems by the JCL PARM field. If you use UTIL0028 to initialize Management Services, the JCL PARM field can be augmented by the UTIL0028 input file—see Appendix C, *The UTIL0028 Utility*, in this guide for further details
- In VM systems by the ATTACH command in the GCS initialization procedure

See the *Unicenter Mainframe Installation and Setup Instructions* for full details.

For all operating systems and methods of input, the format of the parameter information is identical. If you specify multiple parameters separate each with a comma.

### Displaying and Changing JCL Parameter Settings

You can display the current settings of all the JCL parameters with the SHOW PARMS command. To change any of these parameters, specify their new values in the RUNSYSIN member and then restart the system.

For a full list of the JCL parameters available and a description of their use, see Appendix B, *JCL Parameters*.

## Identifying the System to Users

If you have multiple domains or communicate with other systems you should set up your domains so that the users can easily identify which one they are working in. To do this you can set the domain ID and put titles on the panels.

### Identifying Domains and Panels

The NMDID JCL parameter identifies the domain ID for each of your domains. If you have multiple domains, ensure that you have a different domain ID for each one. See Appendix B, *JCL Parameters* for information about the NMDID parameter.

You can use the SYSTEMID (System Identifications) parameter group in Initialization and Customization Services (ICS) to help identify each of your domains. This parameter group specifies a system identifier which is used when you link to other domains. Ensure that each of your domains has a different system identifier.

This parameter group also specifies the titles to be displayed on the logon panel and the OCS console panel. This helps users to identify the domain that they have logged on to.

**Note**

This parameter group takes effect during system initialization only.

See the section, *Customizing Your Management Services Region Using ICS*, on page 2-4, for more information about ICS.

# Customizing Your Management Services Region Using ICS

To perform the tasks in this section, you must have completed the tasks in the *Unicenter Mainframe Installation and Setup Instructions* manual. This section describes the tasks you must complete to customize your Management Services region.

## About Initialization and Customization Services

Initialization and Customization Services (ICS) provides the facility for the review and update of parameter groups.

You use Initialization and Customization Services (ICS) to customize your region. ICS is a computer-assisted initialization facility that enables you to implement a region rapidly and easily. Also, ICS enables you to customize parameters easily at a later stage.

When you first install a product, you need to set various parameters to get the product up and running. ICS helps you set up these parameters. An initial dialog is supplied for the first time user, to walk you through the customization process. You are prompted to supply required parameter values and given the opportunity to supply optional parameter values.

### What Are Parameter Groups?

System parameters are grouped by category (such as Security) in logical parameter groups, to simplify the process of initializing and customizing a region. The parameter groups are listed in Appendix P, *Parameter Groups.*

Groups of individual parameters translate into one or more of the following:

- SYSPARMS that determine how your region functions. (For further information about SYSPARMS, see Appendix D, *SYSPARMS Operands.*)
- Global variables that are used by various NCL applications to control their functions.
- Local parameters that define how to implement actions associated with parameter groups (for example, specifying datasets to be allocated).

## Logging On for First-time Customization

**Note 1**

Before proceeding with the tasks provided in this section, ensure that user IDs have been defined as specified in the *Unicenter Mainframe Installation and Setup Instructions*.

**Note 2**

If you have set up your region using a pre-existing User ID Access Maintenance Subsystem (UAMS) database in which the background users are already defined for your region, you might not be able to action your changes during customization. If this is the case, you must update the background user definitions. Proceed as follows:

1. Enter **$UAMS** at the ===> prompt.
2. Update the background user definitions.
3. Exit back to the customization dialog, and continue the customization process.

As part of the installation and setup tasks, you should have defined an initial user in the region. To perform first-time customization, log on to the region as that user as follows:

Step 1. Enter **LOGON APPLID(***acb-name***)**, using the ACB name of the region.

The logon panel is displayed.

Step 2. Enter the user ID and password. You are prompted to change your password.

Step 3. Change your password.

After your password is changed, the initial System Parameters Customization dialog is presented.

Step 4. Select the Fast Setup Customization Parameters option to review and customize the minimum set of parameter groups.

Fast Setup displays a list of parameter groups that *must* be updated before the System Parameters Customization dialog is complete. This list varies, depending on your combination of products.

To customize a parameter group, proceed as follows:

a. Enter **U** beside it.

b. Press F3 (File) to indicate that you have reviewed the group.

If you update a group, you must press F6 (Action) to action the changes immediately, then press F3 (File) to save your changes to ensure that they are actioned on subsequent restarts of the region.

Step 5. When you finish with the parameter groups, press F3 (Exit) twice.

**Note**

The region has other parameter groups that you can customize. To view the list of parameter groups, enter **/ICS** at the ===> prompt. For further details, press F1 (Help) on any ICS panel.

## Dealing with Initialization Failures

If you log on to a region that has not completed initialization, the Initialization In Progress dialog is displayed to indicate progress and to assist you in identifying and rectifying any problems.

Fatal errors occur (for example, you are unable to log on) if either, or both, of the following are unavailable:

- Panel libraries
- Managed Object Development Services (MODS) control files

**Key Concept**

The MODS control files contain information that is required for the operation of your region. The information includes menus, messages, and online help.

When you are performing customization, it is important to correctly identify all the relevant datasets. You must also ensure that the panel libraries and MODS control files created during installation contain the supplied panels and records.

You can identify the problems by referring to the job log for information related to the PANEL and MODS datasets.

Region startup is also inhibited if there are JCL errors or missing libraries in either your startup JCL or RUNSYSIN member.

### Using the Initialization In Progress Dialog

The Initialization In Progress dialog shows you the current initialization status and whether actions associated with the parameter groups have failed.

The Initialization In Progress dialog enables you to:

- List only failed parameter groups
- List all outstanding parameter groups; that is, those that have failed, are queued to be applied, or are in the process of being applied
- List all parameter groups, including those that have run successfully
- View the messages in the initialization and customization log

The parameter group lists are dynamic action lists. You can apply the following actions to listed parameter groups:

**Note**

An action can only be performed against an already completed parameter group or a failed parameter group.

- **AC** (Action) to action a parameter group
- **H** (Help) to get help about a parameter group
- **I** (Ignore failure) to indicate to the region that it should ignore a failed parameter group and proceed to run dependent parameter groups

  **Note**

  Ignoring parameter groups is not recommended. It might affect the operation of the region. Consider carefully before applying this action.

- **L** (ILog) to view the messages, in the initialization and customization log, that relate to the selected parameter group
- **S**, ***l***, or **B** (Browse) to browse parameter group details
- **U** (Update) to update parameter group details

See the online help for additional information.

When you correct an error by updating an incorrect parameter group record, you have to action that parameter group before processing can continue (unless you apply the Ignore action). Action the parameter group in one of the following ways:

- By pressing F6 (Action) when you finish updating the parameter group
- By applying the **AC** (Action) action after you file the parameter group

**Note**

Some parameter groups can be actioned as soon as you enter appropriate values on the parameter panel. When, however, you change the value of parameters such as the system ID or MODS file names, these parameter values can only be actioned by restarting the system. For further information, see the section, *Updating System Parameters for a Session*, on page 2-9.

## Product-related ICS Parameter Groups

All products supply their own ICS parameter groups in addition to those that apply to Management Services. These additional parameter groups control the customization of the features provided by those products and are not available if the region is not licensed for those products.

## Customizing OCS Function Key Settings

Management Services provides support for up to 24 function keys in OCS mode. You can customize these keys either locally or globally.

### Setting Local Function Keys

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

- Only apply to the requesting panel
- Last only until you exit from OCS

*Example*

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

```
PF4 CONV,MSG USER1+
```

### Setting 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. To do this, execute the PF command in the BSYS environment (for example, in your INIT or READY procedures), or submit a PF command to the BSYS environment. Function keys set in this way apply to all OCS panels unless overridden by a further PF command entered from OCS mode.

For details about the PF command see the *Management Services Command Reference*.

#### Listing Function Key Settings

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

### 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 |

## Updating System Parameters for a Session

Most customization of Management Services is performed by setting system parameters (SYSPARMS) by using ICS.

System parameters are set by using ICS (see the section, *Customizing Your Management Services Region Using ICS*, on page 2-4) or the SYSPARMS command. Each operand of the SYSPARMS command allows you to specify options to change and customize the way Management Services works.

**Note**

Not all SYSPARMS have corresponding ICS parameter groups. It is highly recommended that you use ICS to set all possible SYSPARMS.

For SYSPARMS without a corresponding parameter group, set the SYSPARMS in the INIT and READY procedures initially and then update them by using the SYSPARMS command.

See Appendix D, *SYSPARMS Operands*, for a complete list of all the SYSPARMS operands and a description of their use.

## Displaying and Changing System Parameters

To display a list of the values currently specified for each SYSPARMS command operand, use the SHOW SYSPARMS command. You can also access the current values of the SYSPARMS command operands by using the System Support Services function (see Figure 2-1). To access this panel, enter **/SYSPARM** at the ===> prompt.

*Figure 2-1. Display/Update SYSPARMS Values Panel*

```
PROD------------- System Support : Display/Update SYSPARMS Values------------
 Command ===>

 SYSPARM  Stat  Value                    SYSPARM  Stat  Value
 ABENDCD        4095                      ACBRETRY       YES
 ACCDFLT        ALLOW                     AOMCMDCH       >
 AOMCMIGI       YES                       AOMCMIGX       NO
 AOMCOPTS       00                        AOMCPREF       DE1N
 AOMCTO1        10                        AOMCTO2        20
 AOMCTO3        5                         AOMCTYPE       EXTMCS
 AOMCUTOK       YES                       AOMDQLIM       1000
 AOMGDBSL       16000                     AOMGQFL        3
 AOMGQSZ        500                       AOMGTHRS       5
 AOMJESCH       $                         AOMLCMD        NO
 AOMLOG         NO                        AOMMAXTS       4096
 AOMMLTO        5                         AOMMPFEV       GNN
 AOMMPFGN       GNN                       AOMMPFIA       WNH
 AOMMPFOC       TNN                       AOMMPFOD       GNN
 AOMMPFOL       TNN                       AOMMPFPP       GNN
 AOMMPFUA       RNH                       AOMPQFL        3
 AOMPQLIM       32767                     AOMPQSZ        10000
 AOMPRFJI       YES                       AOMPRFJN       NO
 AOMPRFMN       NO                        AOMPRFSN       NO
 AOMPRFTM       HMS                       AOMPRFUI       YES
```

To change the values of any operand, type over the displayed value with the new value required. The changes last for the current session only.

To have these SYSPARMS values available across system restarts, update your INIT member or the relevant ICS parameter group (enter **/ICS** on any panel).

## Customizing Management Services Options

To specify an option with the SYSPARMS command, enter the command at the OCS command line. The format of the SYSPARMS command is:

`SYSPARMS operand=`*`value`* `operand=`*`value`* `...`

*Example*

To display the time at the end of the OCS title line, enter the following command:

```
SYSPARMS OCSTIME=YES
```

## Initialization Options

There are some SYSPARMS command operands that cannot be changed while the system is operational. These operands must be included in your INIT procedure so that they are executed during initialization. Some of these operands are:

- GENRSRC
- IPCHECK
- LANG
- MAIFILE
- MDFYLIM
- NCLTRLFF
- PWEXPIRE
- PWMAX
- PWMIN
- PWRETRY
- SYSCONNM
- SYSCONSO
- SYSCONUI
- SYSCONXU
- VTAMCDRM
- VTAMNTFY

If you specify new values for these operands, the new values do not come into effect until the system is initialized. All other operands can be changed during system operation by authorized users.

See Appendix D, *SYSPARMS Operands*, for a complete list of all the SYSPARMS operands and a description of their use.

## Restricting SYSPARMS Command Usage

SYSPARMS command usage can be restricted in the same way as other Management Services (MS) commands.

You can use the INIT processing to change the authority level for the SYSPARMS command to a level higher than that assigned to any user ID. This prevents use of the command except during initialization processing.

The CMDAUTH operand of the SYSPARMS command, which allows the authority levels of all other commands to be changed, requires a user command authority level of nine unless otherwise changed during initialization processing.

**Caution**

Avoid changing authority levels, because ICS and other system functions can be affected if the background system user (BSYS) is not authorized to issue SYSPARMS or other commands.

### Product-related SYSPARMS Command Operands

Other products supply their own SYSPARMS command operands in addition to those that apply to Management Services. These additional operands control the customization of the features provided by those products and are not available if the system is not licensed for those products.

See Appendix D, *SYSPARMS Operands*, for information about product-specific SYSPARMS command operands.

## Using 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 ICS to set the various console parameters.

### Allowing Dynamic Determination of the Need for Migration IDs

You can allow dynamic determination of the need for a migration ID in two ways:

- 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).

  For further details of the SYSCMD command and its related SYSPARMS, see Appendix D, *SYSPARMS Operands*.

- 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 further details of the &AOMMIGID built-in function, see the *Network Control Language Reference* manual.

**Note**

If the current console type does not support or need migration IDs, then 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.

For further information about user migration ID exits, see Appendix P, *Defining User Migration ID Exits*.

## Customizing Command Authority Levels

Management Services commands and VTAM commands that Management Services can execute, are each assigned a default authority level. These are within the range of 0 to 255. See Appendix H, *Command Authority Levels*, for a list of the default authority levels assigned to commands.

Users of Management Services are also allocated a command authority level. This is in their user ID definition. The authority level set corresponds to the authority level for the commands they are authorized to issue. For information on allocating command authorities to users, see Chapter 3, *Implementing Security*.

### Changing Command Authority Levels

To allocate a different command authority to a particular command, use the SYSPARMS CMDAUTH operand.

*Example*

To allocate a command authority of 5 to the CANCEL command, enter the following command:

```
SYSPARMS CMDAUTH=(CANCEL,5)
```

## Disabling Commands

You can disable a command by allocating an authority level higher than that available to any user.

## Replacing Commands with NCL Procedures

To replace native commands with an NCL procedure of the same name, use the CMDREPLS parameter group in ICS.

*Example*

To run an NCL procedure in place of the CANCEL command, do this:

Step 1. Enter **/ICS** at the ===> prompt.

The ICS : Customization Parameters panel is displayed.

Step 2. Enter **U** beside the CMDREPLS parameter group.

The ICS : Initialization Parameters panel for the CMDREPLS parameter group is displayed.

Step 3. Type **CANCEL** in the next available position in the Replaced Command ID field.

Step 4. Press one of the following function keys:

- F3 (File) to file the changes—the changes are not applied to the current region, but are applied when the region is restarted.
- F6 (Action) to apply the changes to this region—the changes are not saved.

## Equating 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 ROF or MAI. You can also use the EQUATES parameter group to replace an existing command with an NCL procedure.

You can set up equates in the EQUATES parameter group when you first customize your Management Services 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.

See the *Management Services Command Reference* for more information about the EQUATE command.

### Replacing Commands After System Initialization

To substitute commands with NCL procedures after initialization, you can use either of the following:

- The CMDREPLS parameter group
- The EQUATES parameter group

**Note**

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

## Implementing the Timeout 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 timeout facility is provided by Management Services. Use this to specify timeout intervals and actions for all terminals.

The timeout facility applies to Management Services users. It 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 timeout management is implemented, it affects all of the users of your Management Services system, unless their user ID definitions exempt them.

Terminals under EASINET control are not affected by the timeout facility. This is because there are no individual users associated with such terminals.

## Two-level Timeouts

Having a two-level timeout system provides greater flexibility in timeout management. When the first-level interval expires, you can warn the users that they will be canceled when the second-level interval expires, unless some activity takes place on the terminal.

If the first timeout period action disconnects or cancels the user, then specifying a second timeout serves no purpose. This is because the user is no longer logically available to accrue a timeout interval.

## Implementing Timeouts

Timeout can be implemented in two ways:

- System level—using SYSPARMS timeout parameters (see the following sections for details)
- User level—by specifying values in user ID definitions (see Chapter 3, *Implementing Security*, for details about specifying user ID definitions)

## Specifying System-level Timeout Intervals

The system-level timeout facility provides two timeout intervals. These intervals are best thought of as two separate clocks maintained for each terminal where a user is logged on. The first-level timeout clock starts as soon as a user logs on to Management Services from any terminal. The interval is decremented until it expires, or until user activity at the terminal resets the clock.

If the first-level timeout interval expires, the second-level timeout clock for the terminal starts. The second-level timeout interval is decremented until it expires, or until further activity takes place at the terminal.

The interval for each timeout level is specified in minutes and seconds within a range of 1 second to 999 minutes 59 seconds in the format *mmm.ss*.

### Procedure—Setting the First-level Timeout

To specify the first-level timeout, enter the following command:

```
SYSPARMS TOTIME1=mmm.ss
```

### Procedure—Setting the Second-level Timeout

To specify the second-level timeout, enter the following command:

```
SYSPARMS TOTIME2=mmm.ss
```

*Example*

To set a first-level timeout of ten minutes and a second-level timeout of 30 seconds, issue the following commands:

```
SYSPARMS TOTIME1=010.00
SYSPARMS TOTIME2=000.30
```

## Specifying System-level Timeout Actions

The timeout actions that you specify determine the system action that occurs when either of the timeout intervals expires.

### Procedure—Setting the First-level Action

To specify the action the system takes when the first-level timeout interval expires, enter the following command:

```
SYSPARMS TOACT1=action
```

### Procedure—Setting the Second-level Action

To specify the action the system takes when the second-level timeout interval expires, enter the following command:

```
SYSPARMS TOACT2=action
```

### Types of Timeout Action

Timeout actions can be one of the following:

**LOG**
Record the timeout on the activity log.

**ALARM**
Ring the alarm at the terminal to warn the user.

**LOCK**
Display the LOCK panel and require the user to re-enter their password before continuing.

**DISC**
Disconnect the user, leaving a reconnect region.

**CANCEL**
Cancel the user.

## Repeating Timeout Actions

You can use the SYSPARMS TOACT1 and TOACT2 operands to specify a repeat interval as well as the action code. The repeat interval represents the interval after which the timeout action is repeated if the user remains inactive once the initial timeout action is taken. It is specified in minutes and seconds. The repeat interval is valid only if the action was LOG or ALARM.

If a repeat interval is specified on the first-level timeout, the first-level action repeats until terminal activity resets the timeout clock, or the second-level timeout interval expires.

## Specifying the Timeout Trigger Event

The SYSPARMS TOSTART operand specifies the timeout trigger event that resets the timeout interval for the terminal, and starts the clock again.

The options are:

**KBUNL**
Keyboard Unlock. The timeout interval is reset and the timeout clock started again for each terminal, each time its keyboard is unlocked and user input is allowed.

**INPUT**
The timeout interval is reset and the timeout clock started again for each terminal whenever any input is received from it.

**OUTPUT**
The timeout interval is reset and the timeout clock restarted for each terminal whenever output is sent to it.

**TIO**
The timeout interval is reset and the timeout clock started again for each terminal whenever any input is received from it, or whenever output is sent to it.

## Starting the Timeout Facility

To start timeout management and specify the trigger event, enter the SYSPARMS TOSTART command.

It is possible to override the timeout facility for specific users in their user ID definition. See Chapter 3, *Implementing Security*, for details on how to define a user ID.

## Starting the Timeout Facility Automatically

To have the timeout facility set automatically, include the SYSPARMS command operands for the default timeout intervals and action codes in your INIT procedure. This is executed automatically during system startup. Include the following commands:

| | |
|---|---|
| SYSPARMS TOTIME | Set the first-level timeout interval |
| SYSPARMS TOTIME2 | Set the second-level timeout interval. |
| SYSPARMS TOACT1 | Specify the action to be taken on expiry of the first-level timeout interval, and optionally a repeat interval. |
| SYSPARMS TOACT2 | Specify the action to be taken on expiry of the second-level timeout interval, and optionally a repeat interval. |
| SYSPARMS TOALARM | Set the number of times the alarm is to ring if ALARM is the action to be taken on timeout interval expiry. |
| SYSPARMS TOSTART | Start timeout management and specify the timeout trigger event. |

## Changing Timeout Parameters Dynamically

The various timeout parameters can be changed at any time by suitably authorized OCS users. Although it is unlikely that parameters such as the timeout actions will need to be changed online, timeout intervals are commonly:

- Modified during the day according to operations requirements
- Only required between certain hours of the day

The SYSPARMS command provides the capability to modify the timeouts. You can use the AT command to issue this automatically at a nominated time.

### Suppressing Timeout Activity

An interval of 0.00 (zero seconds) for the first-level timeout stops all timeout management. An interval of 0.00 for the second-level timeout suppresses all second-level timeout activity.

Timeout management resumes when a TOTIME1 command is entered, specifying a positive time interval.

### Other Timeout Considerations

When planning timeout implementation, consider the following factors:

- All users are subject to timeout except those with UAMS definitions which exempt them (operators for example).
- The two-level timeout facility is not mandatory. A zero time interval specified for the second-level timeout (TOTIME2) indicates that no second-level timeout applies. If the timeout action is for disconnection or cancellation it is more user-friendly to use the two-level system. In this case the first-level timeout is used to warn the user that their first timeout period has expired (TOACT1=ALARM) and the second-level timeout disconnects or cancels them (TOACT2=CANCEL).

## Customizing National Language Character Set Support

Management Services 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. See the appendix *Supported Language Codes for National Language Support* in the *Network Control Language Reference* for a list of supported language codes and their associated code pages.

You can set an individual language code for users in their user ID definitions. See Chapter 3, *Implementing Security*, for information on how to specify a language code in a user ID.

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—#FLD CAPS=YES
- NCL statements—&TRANS, NLUPPER, and NLLOWER operands

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

- &NDBSCAN—scan expressions that 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

### Determining 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

## Customizing Management Services for Use with DBCS

One class of 3270 terminal is used to enter and display the text of languages that use symbols instead of individual letters (for example, Japanese language Kanji symbols). The complexity and number of symbols in these languages require that two bytes be used to represent each symbol. These terminals use double byte character streams, or DBCS.

Handling datastreams for such devices is significantly different from normal processing, so Management Services operates differently when DBCS devices are connected to the system.

## Setting up Management Services to Support DBCS

To set up Management Services so that your network uses DBCS devices, include the DBCS parameter in your JCL PARMS.

The difference between the various options for this command is explained below in the section titled *IBM and Fujitsu DBCS Implementation.*

If your system supports DBCS, there is no longer any concept of upper and lower case letters—DBCS support treats every character as being unique, so ‘a’ and ‘A’ are regarded as two different characters, not as two versions of the same character. This means that OCS commands must be entered in upper case letters only. The operation of some NCL statements changes because the upper case translation implicit in Single Byte Character Stream (SBCS) operation does not occur in DBCS operation. The *Network Control Language User’s Guide* identifies all statements that are affected.

## Identifying DBCS Devices

Management Services determines whether a terminal is DBCS-capable from the BIND information response as the terminal is connected. The actual notification procedure varies according to the type of device.

IBM devices indicate they are DBCS-capable by responding to a QUERY issued by Management Services once the BIND parameters indicate the terminal supports the QUERY command. If you use the wrong BIND parameters, the system does not issue QUERY and will not treat the device as a DBCS terminal.

Fujitsu terminals indicate explicitly in the BIND parameters that they are DBCS-capable. If you use the wrong BIND parameters, Management Services will not recognize the terminal’s DBCS capability. See Appendix A, *Connecting Terminals to Management Services*, for examples of BIND parameters that you should use for DBCS support.

## Specifying a DBCS Device Dynamically

To test DBCS devices, use the TERMINAL command. This lets you dynamically specify that your terminal is to be treated as a DBCS-capable device of either IBM or Fujitsu implementation.

## IBM and Fujitsu DBCS Implementation

DBCS devices can operate in both DBCS and SBCS modes They differentiate between the two operating modes by shifting in and shifting out of DBCS mode. The shift in and shift out indicators take the form of attribute bytes within the terminals datastream. The hexadecimal code used to represent these bytes differs in the IBM and Fujitsu implementations. This is handled automatically by Management Services.

The IBM and Fujitsu implementations of Japanese Kanji language symbols use different hexadecimal code points. This means that the hexadecimal representation for a particular symbol on an IBM device is different on a Fujitsu device. A screen panel for an IBM terminal with Kanji characters may not make sense if displayed on a Fujitsu terminal.

## DBCS Text Strings

Anomalies occur if DBCS text strings exist in databases allocated to Management Services. Management Services cannot tell whether the DBCS characters are for IBM or Fujitsu formats unless specifically told what to look for. To accommodate these differences, the DBCS JCL parameter has the following values:

**IBM**

Regardless of the operating system, Management Services assumes that any external (file) data containing DBCS strings uses IBM code points. Fujitsu terminals are still recognized and treated correctly.

**FUJITSU**

Regardless of the operating system, Management Services assumes that any external (file) data that contains DBCS strings uses Fujitsu code points. IBM terminals are still recognized and treated correctly.

**YES**

Management Services assumes a value of IBM if running under an IBM operating system, or a value of Fujitsu if running under MSP.

In an installation using mixed operating systems (for example, MVS and MSP), these operands allow systems running under the different operating systems to operate in the same manner.

**Note**

The UDBCTL OPEN command lets you override the default as specified by DBCS JCL parameter for individual files.

# 3

# Implementing Security

This chapter provides information about implementing security for users of Management Services.

**This chapter contains the following topics:**

- About Security
- Implementing Security for the First Time
- Defining Users to the System
- Defining a User ID
- Defining a Model User ID
- Defining a System Console User ID
- Defining Background Environment User IDs
- Accessing User ID Definitions Using NCL
- Providing a Customized Primary Menu
- Defining Libraries and Paths

## About Security

Security for Management Services can be controlled using one of the following:

- The User ID Access Maintenance Subsystem (UAMS)
- An external security exit, that operates in conjunction with UAMS or completely replaces it; options include:
  - A combination of UAMS and a supplied SAF security system interface that performs user ID and password validation
  - A combination of UAMS and a user-written security exit that performs user ID and password validation
  - A user-written security exit that performs all security functions

Below are descriptions of each security option. These descriptions can help you to choose the best option for your installation.

**Note**

Each region can be independently set up to use a different security option, if required.

### UAMS

UAMS is designed to provide a fully self-contained security system. It allows you to define user IDs for each user of Management Services. User IDs provide logon and password checking and can be added, deleted, or updated.

You can use UAMS to define:

- Each user ID separately
- A group ID to be used as a model for each user that requires similar access and authority

UAMS by itself is independent of any external security system. Passwords stored in UAMS are not synchronized with RACF, for example.

A single UAMS dataset can be shared by any number of regions.

## External Security Exit

An external security exit can be used to provide partial or full security processing:

- A *partial* security exit *supplements* UAMS by replacing the password checking part of UAMS with external security system validation of the user ID and password. Other user profile information is still maintained on UAMS, although the exit has the option of supplementing or overriding this information.
- A *full* security exit *replaces* UAMS. All security and user profile information must be supplied by the exit.

Each type of external security exit is explained in more detail in the following sections.

Source code for sample exits is distributed with Management Services. These include a SAF partial exit, and RACF and ACF2 full exits. These are supplied as-is, to show how an exit should be written.

**Note**

Using any form of security exit can have ramifications on some products; for example, on system user IDs. These issues are explained in more detail in the following sections.

### Partial Security Exit

If your region operates with a partial security exit, then UAMS password checking functions are disabled. Instead, the exit is called to validate a user ID and password. Typically, this is done with a call to the external security system (for example, RACF).

The exit can also supply overriding or additional user profile attributes. The exit can also control modelling, whereby users can be dynamically created in the UAMS dataset the first time that they use Management Services.

### Supplied Partial Security Exit

You can use a supplied partial security exit. If the JCL parameter SEC=PARTSAF is coded, then your region operates with a SAF partial security exit. This exit performs straightforward processing; for example, for logons. It requires the user to be defined to the external security system and validates the password. If a model name is set (by using the SYSPARMS MODLUSER command), and the user is not known to the region, then the user is defined to the region and the nominated model user ID is used to build the user's profile in UAMS.

## Full Security Exit

If your region operates with a full security exit, then no UAMS dataset is used. The security exit must perform all required security functions; specifically it must provide all information about user authority on the region.

## Considerations When Using a Security Exit

Some NetMaster, NetSpy, and SOLVE products may not operate correctly if a security exit is in use. This is because many products make use of system users to perform work. A system user is an internal user, automatically logged on. System users do not correspond to any real user. The internal logon occurs even if the security exit says that the user is not defined. These users can log on to other regions. However, when this occurs, the target region's security exit attempts to validate the user.

Thus, if you are using a security exit, these user IDs (that log on to other regions) must be defined to the external security system. However, no specific password is required, because the validation call simply checks that the user is known.

If you have many regions that interconnect, then, by using the NMSUP JCL parameter, you can reduce the number of unique user IDs that must be defined this way. Set the value of NMSUP for all the regions to the same value (for example, NM01). The system user IDs in each region will then have the same names (for example, NM01BSYS, NM01BLOG, NM01AOMP).

If the NMSUP prefix is not specified, the domain ID (NMDID) is used. If the domain ID is not specified, the first characters (up to a maximum of 4) of the system primary ACBNAME, specified in the PRI parameter (or the default if not specified), are used as the default value.

**Note**

Each domain has a unique domain ID (NMDID). This means that letting NMSUP default results in a unique NMSUP value for each domain. Therefore, if you want to have a common system user prefix across background user IDs, you need to explicitly specify a value for the NMSUP JCL parameter.

See Chapter 4, *Implementing Additional Security*, in this guide, for detailed information on the types of access security exits that are supported.

## Implementing Security for the First Time

To implement security for the first time, you must first log on to the product. Use INSTALL as the user ID and 99999999 as the password on the logon panel, as described in your *Unicenter Mainframe Installation and Setup Instructions*.

**Note**

If you are using an external security package instead of UAMS, you need to define appropriate assembler exit routines and user IDs for this product before it is possible to log on.

You must now define an initial set of user IDs, including your own, making sure one of them has UAMS authority. See the following section, *Defining Users to the System*, for details on how to do this.

## Defining Users to the System

There are different types of user ID that can be defined to your system:

- Individual user
- Model
- System Console
- Background Environments
  - Background process
  - System procedure

The following sections describe how to define each type of user ID listed above.

## Defining a User ID

Before anyone can access Management Services, they must be defined as an authorized user. A one to eight character user identifier (the user ID), together with a password, is used to associate an individual to the privileges and authorities allocated them. The password can be selected and maintained by the user, or an external security system.

In addition to identifying an individual user of the system, the user ID also defines the following information about a user:

- Statistical information—personal information such as name, location, telephone number, as well as user ID expiry date and start times of last session.
- Control information—identifies the functions the user is allowed to perform and the features they can access when logged on to the system. This information determines, for example, whether a user is allowed to act as an OCS operator and if so, their authority level.

### Procedure

To define an individual user ID for a new user, complete the following steps:

Step 1. Enter **/UAMS** at the ===> prompt.

The UAMS : Primary Menu is displayed.

Step 2. Type **A** at the Select Option ===> prompt and press TAB.

Step 3. Type in the User Name, Definition Type, and Confirm Delete? values into their fields and press ENTER.

The first of several panels of user ID definition details is displayed. These describe a user's access to various features.

Step 4. Type the required information on each following panel, scrolling forward (F8) to review the next panel.

Each UAMS panel is described in detail in Appendix E, *UAMS Panel Descriptions*.

Step 5. Press F3 (File) to save the new user ID definition when all the panels have been reviewed and the required attributes specified.

You are returned to the UAMS : Primary Menu.

**Note**

If the user ID you are defining is similar to another ID, you can save time by copying an existing ID to a new user you wish to add. Do this by selecting option C (Copy) from the primary menu, then specify the user ID for the existing user as the model. UAMS retrieves the details for that user ID and then enters Add mode. You must now enter the new user ID name and change any fields required, before filing the new definition under the new name.

## Maintaining User IDs

You can be obtain a selection list of the user IDs defined to your system. The list allows you to Browse, Force Password, Update, Delete, or Copy any of the listed user IDs. If you are not authorized for UAMS maintenance functions, only the browse function is allowed.

The following information is displayed about each user ID:

- Name
- Location
- Telephone number
- Type (Group or User definition)
- Access privileges

### Procedure

Step 1. Type **L** at the Select Option ===> prompt on the UAMS : Primary Menu and press TAB.

Step 2. Delete the contents of the User field and press ENTER

An alphabetic list of all user IDs is presented.

Step 3. To obtain a partial listing, enter a prefix in the User field to list only those user IDs beginning with that prefix.

Step 4. To display the command authority and access privileges information press F11 (Right).

## Forcing a Password Change for a User ID

If a user forgets their password, you can allocate a new password using the Force User's Password Change facility.

### Procedure

Step 1. Enter **/UAMS** at the ===> prompt.

The UAMS : Primary Menu is displayed.

Step 2. Type **F** at the Select Option ===> prompt on the UAMS : Primary Menu, change the User field to the user ID you want to change the password for, and press ENTER.

The UAMS : User Details panel for the specified User ID is displayed.

*Figure 3-1. UAMS : User Details Panel*

```
USER01------------------- UAMS : User Details --------------------Page 1 of 1
Command ===>                                                  Function=Force

User ID ...................... USER01     Last Logon . WED 04-JUL-2001 10.01
New Password ................            Last Updated ... 19-JUL-2000 10.14
                                         Definition .....

User Name .................... User One
User Location ................ Sydney
Phone Number ................ Ext 369









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

Step 3. Type the new password in the New Password field, and press ENTER.

The message, `PASSWORD VERIFICATION COMPLETE`, is displayed.

Step 4. Press F3 (File) to save the changes.

When the user next logs on to Management Services they are prompted to change the forced password. If a security exit is provided for password processing, this option can be suppressed by the exit.

**Note**

This function is not available from the special Install User ID, or for Group user IDs.

## Defining a Model User ID

Many installations have numerous users who access only one or two functions (for example, those users whose only privilege is to use SOLVE:Access). Defining and maintaining system access for individual users requires considerable administrative effort. To minimize this effort, you can set up a model user ID so that users can automatically log on and register.

A model user ID is defined in the SYSPARMS MODLUSER operand. It has the following format:

```
SYSPARMS MODLUSER=userid | NONE
```

See Appendix D, *SYSPARMS Operands*, for a detailed description of the MODLUSER operand.

### How It Works

If no security exit is in place and a model user ID is defined, a user who tries to log on with an undefined user ID, but using the password of the model user ID, causes the following to occur:

- The model user ID definition is read
- A new user ID is created using the user ID specified during the logon request with the attributes of the model user ID
- The new user ID is saved and flagged as a new user
- The user ID creation is recorded on the log
- Logon is allowed to proceed
- The new user is prompted to change their password and fill in personal details (name, telephone, location, and so on) before the logon is complete

**Note**

Creating new user IDs from model definitions is not suitable when defining high authority user IDs.

If no security exit is in place, and no model user ID has been defined, logon attempts from user IDs that are not defined are rejected.

If a model user ID has been specified in the SYSPARMS MODLUSER operand but has not been defined to UAMS, logon requests by undefined users will fail as if no model was defined.

If a security exit is in place, model user IDs work differently. See *User ID Modeling*, on page 4-5.

## Defining a System Console User ID

The system console needs a special type of user ID. This is because:

- It only receives messages
- It has no full screen capabilities
- User logon is not possible

The system console user ID can be defined in the same manner as any other user ID, however, only fields that are applicable to message receipt are valid. For example, by defining the system console user ID as a monitor status OCS operator with PPO authority and an appropriate command authority, the system console can be profiled as a fully functional OCS operator console in the same manner as any other user ID.

The following sections describe how the system console user ID is created. See the section that applies to your operating environment.

### Defining the System Console User ID in OS/390 or Later Environments

The OS/390 or later environment supports named consoles as well as extended MCS consoles, RACF signon, and security for consoles. The main consideration is that systems can receive console names that do not have a 1-byte console ID.

If necessary, the console user ID can be reproduced in UAMS so that a user who has limited authority cannot circumvent that authority by going to a console and issuing a MODIFY command.

#### Default OPER Environment

A default CONSOLE environment allows messages to be delivered to the operator. These messages are then delivered using the routing and descriptor codes set by the SYSPARMS ROUTCDE and DESC operands. This environment is built after INIT has finished.

The default terminal name for the system console environment is CONSOLE. The user ID for the console is automatically assigned using the following process:

1. The value of the SYSPARMS SYSCONUI operand is examined
2. If no value is defined, it looks at the default—*pppp*OPER
3. If there is no definition for *pppp*OPER, the system assigns .DFLTOP as the user ID

If .DFLTOP is used because no other value is defined, problems with ROF routing to other systems might result. Even if a definition for SYSOPER exists from an earlier version, the new default operator name will mean that it is no longer used.

See Appendix D, *SYSPARMS Operands*, for detailed information about the SYSPARMS DESC, ROUTCDE, and SYSCONUI operands.

## Actual Console Environments

A system console environment is signed on the first time that a command (for example, MODIFY) is sent from the console to the system.

The terminal name used is one of the following:

- The MVS console name if:
  - SYSPARMS SYSCONNM=ALL is in effect
  - It is an extended MVS console
- CONS#*nn* or CONS#*nnn*.

The user ID is determined by the values of the SYSPARMS SYSCONSO, SYSCONUI, and SYSCONXU operands as follows:

- SYSCONUI—sets the basic user ID. If not specified, it defaults to *pppp*OPER.
- SYSCONSO—determines the search order for user IDs when a user attempts to sign on to the console
- If SYSPARMS SYSCONXU=NO is in effect, the user ID is determined as shown in Table 3-1 on page 3-12, with one exception. The extended MCS consoles use the SYSCONUI user ID value as the logical user ID. This is because a user ID cannot be derived from the console ID. If a SAF UTOKEN is not supplied, then the information in Table 3-1 applies. This is because a non-master console is not signed on and the user ID is not known.
- If SYSCONXU=YES is in effect, the user ID is determined as shown in Table 3-2 on page 3-12.

When an attempt to log on to the system console is made, the system tries three times to assign a user ID to the system console. The outcome depends on the value of the SYSCONSO operand. Table 3-1 and Table 3-2 (depending on the value of SYSCONXU) describe this process.

*Table 3-1. Assigning a System Console User ID (SYSCONXU=NO)*

| | **Attempt 1** | **Attempt 2** | **Attempt 3** |
|---|---|---|---|
| **SYSCONSO= DEFAULT** | User ID is set to *pppp*CN*xx* or *pppp*C*xxx* | User ID is set to the value of SYSCONUI | User ID is set to .DFLTOP |
| **SYSCONSO= NO** | User ID is set to the value of SYSCONUI | User ID is set to .DFLTOP | |
| **SYSCONSO= REQUIRED** | User ID is set to *pppp*CN*xx* or *pppp*C*xxx* | | |

*Table 3-2. Assigning a System Console User ID (SYSCONXU=YES)*

| | **Attempt 1** | **Attempt 2** | **Attempt 3** |
|---|---|---|---|
| **SYSCONSO= DEFAULT** | User ID is set to the RACF user ID signed on at the console | User ID is set to the value of SYSCONUI | User ID is set to .DFLTOP |
| **SYSCONSO= NO** | User ID is set to the value of SYSCONUI | User ID is set to .DFLTOP | |
| **SYSCONSO= REQUIRED** | User ID is set to the RACF user ID signed on at the console | | |

The console is signed on by trying each attempt in turn until one succeeds.

See Appendix D, *SYSPARMS Operands*, for more information about the SYSPARMS SYSCONUI, SYSCONSO, SYSCONXU, and SYSCONNM operands.

When a RACF-user ID is signed on at the system console, there are 2 special cases, as follows:

- If a user is not signed on at the master console, RACF uses an internal name of *BYPASS*. This defaults to .MASTOP in this case.

  If .MASTOP is encountered, the signon always succeeds, with system assigned defaults.

- If the user is not signed on for other consoles, an internal user ID of .NOTSIGN is used.

  If .NOTSIGN is encountered, the signon of that user ID fails, leading to a try of the next user ID, and so on. If it is the last, the signon fails completely.

### Receiving Command Replies on the System Console

All commands entered at the system console in an MVS system are treated as private to that console. The results of the commands entered are returned only to that console.

## Defining the System Console User ID in VM Environments

The system console for a VM environment is created during system initialization. The logical terminal name is CONSOLE. The system console is used as a target to deliver messages to the operator.

The system console is automatically signed on during system initialization (after INIT has finished). The user ID for the console is automatically assigned using the following process:

1. The value of the SYSPARMS SYSCONUI operand is examined
2. If no value is defined, it looks at the default—*pppp*OPER
3. If there is no definition for *pppp*OPER, the system assigns .DFLTOP as the user ID

If .DFLTOP is used because no other value is defined, problems with ROF routing to other systems might result.

See Appendix D, *SYSPARMS Operands*, for detailed information about the SYSPARMS SYSCONUI operand.

**Note**

These environments do not support multiple operating system consoles.

## Using ROF with System Consoles

System consoles can establish ROF sessions with remote domains.

If a user ID has been defined to UAMS for a specific console, then a corresponding user ID must be defined on every other domain to which the console user ID can establish a ROF connection.

If no specific console user ID has been defined and the console is operating with the same privileges as defined for the console user, the console user ID may establish ROF connections to any other domain without specific user ID definitions being required by the other domains. The console user ID uses the ROF attributes of the console user instead.

## MSGPROC and System Console User IDs

The user ID environment for a system console can have a standard MSGPROC associated with it. MSGPROC processing is activated automatically during the console's signon.

## Unsolicited Output to the System Console

To have the system console receive unsolicited messages, for example, PPO messages or Monitor class messages, direct them to a console user ID. The default system routing codes then determine which physical consoles receive the messages.

For example, if Management Services receives PPO messages and has no one to report them to, the messages are automatically sent to *pppp*OPER so that they will be seen on the system console. It is the system routing codes (as set by the SYSPARMS ROUTCDE operand) that then determine which consoles receive the messages.

You must ensure that the system console routing codes applying in your installation allows Management Services to route PPO messages successfully to the system console if no PPO authorized users are logged on as native Management Services users.

If at least one signed-on console is profiled as a PPO receiver, then messages are regarded as deliverable, and are not sent to the console user automatically.

## Defining Background Environment User IDs

There are two types of background environments:

- Background processes which include:
  - BMON—background monitor
  - BLOG—background logger
  - BSYS—background system process
- System procedures—which execute in special system-level environments logically signed on before the procedure starts. Examples include:
  - LOGPROC
  - PPOPROC
  - AOMPROC
  - CNMPROC

### Initialization User IDs

Each background environment is assigned a special user ID by the system at initialization.

**Note**

Background environments cannot be cancelled.

These user IDs are formed using the system user prefix as defined in the NMSUP initialization parameter. For example, if your system has a system user prefix of NM01:

- The background environment user ID is defined as NM01BMON
- The LOGPROC system procedure user ID is defined as NM01LOGP.

**Note**

If this prefix is not specified, the domain ID (NMDID) is used. If the domain ID is not specified, the first characters (up to a maximum of 4) of the system primary ACBNAME, specified in the PRI parameter (or the default if not specified), are used as the default value. For descriptions of the NMDID and NMSUP JCL parameters, see page B-5.

To see the names of these processes on your own system, enter a SHOW USERS command to list background environment users.

## Initialization Privileges

When the system initializes, the background environment users are logically signed on. If a UAMS user ID is defined for a background environment, the attributes and privileges for it are determined from the user ID definition. If no user ID is defined, the system assigns the background environment with the following privileges:

- Time zone of the system
- Maximum command authority
- OCS authority

## Limiting Background Environment Authorities

To limit the authority of background environments in a system, define the appropriate user IDs to UAMS.

To restrict the NCL procedures that are available to background environments, specify an alternate NCL library in the user ID definition.

## Using ROF with Background Environments

Background environments can have ROF sessions with connected domains. Background environment's need to have their user IDs defined to all of the remote domains that they will to log on to.

## MSGPROC and Background Environment User IDs

Background environment user IDs can have standard MSGPROCs associated with them. To associate a MSGPROC with a background environment user ID, define the user ID explicitly to UAMS and have a MSGPROC included in that definition.

## Accessing User ID Definitions Using NCL

The NCL verb &SECCALL allows you to access the entire contents, privilege levels and attributes of any nominated user ID defined to UAMS. The &SECCALL GET statement retrieves a nominated user ID and presents the requested user ID information to an NCL procedure.

### Accessing User ID Information

You can access all the information about a specified user ID or you can specify the structured fields that identify particular fields within the user ID. The following syntax is used to obtain this information:

```
&SECCALL GET USERID=userid FIELDS=(nnnn,...,nnnn)
```

Each field that is retrieved from the user ID is given a default name generated as follows:

```
&SECnnnn
```

where &SEC is the default prefix, and *nnnn* is a 4-digit number corresponding to one of the structured field keys used to identify fields within the user ID.

*Example*

The structured field key used to identify the user ID name is 0010. When this field is retrieved, it has a name of &SEC0010.

See Appendix G, *Structured Fields*, for more information about structured fields.

**Note**

Only those user ID privileges relevant to the configuration of the system in which the &SECCALL GET statement is executed can be retrieved.

The &SECCALL GET statement operates in the same manner if a full security exit is implemented that supports the relevant calls generated by these statements.

## Providing a Customized Primary Menu

If you are going to customize the primary menu for your installation, you need to understand how the supplied primary menu procedure works.

**Note**

If you are going to customize the supplied procedure, ensure that you rigorously test any changes for all class of user before implementing the new procedure.

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

### About the 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. The $NMPEXIT procedure is discussed below.

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

A reconnection menu is also supplied by $NMPMENU. This provides multiple-signon users with a choice of reconnection options after a session outage. The following choices are provided:

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

SYSPARMS RECONN=YES must be specified if a user is to be presented with the reconnection menu.

## Displaying 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 therefore display on the menu.

The following options and their method of invocation are displayed in Table 3-3.

*Table 3-3. Primary Menu Options*

| Option | Description | Method of Invocation |
|---|---|---|
| U | User Services | Executes the name of the NCL procedure define for the user—see structured field X'0028' |
| O | Operator Console Services | SETMODE OCS |
| M or MAI | Access Services (see first note) | SETMODE MAI |
| T | Data Transfer Services (see second note) | Executes FTS command |
| X | Terminate | — |
| | All other options | Either displays a CAS menu or executes an NCL procedure |

**Note**

Access Services provides entry to the Multiple Application Interface component of SOLVE:Access. This option is applicable only if your region is configured with the SOLVE:Access product. The menu option will be one of the following:

- M—if the region is not configured with any product that uses option M for Monitors (network management and operations management products)
- MAI—otherwise

**Note**

Data Transfer Services provides entry to the File Transmission Services product. This option is applicable only if your region is configured with File Transmission Services.

## Altering 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 the user exits OCS, they are returned to the primary menu, and the primary environment operates in base NCL mode.

## Supporting 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.

## Defining Libraries and Paths

During the installation of Management Services, two panel libraries are created:

- PANLDIS
- PANLUSR

The PANLDIS file is populated by the installation process. The PANLUSR file is for your own use. In addition to these files you can create additional panel files. If you create your own panel definitions, they need to be stored in a panel library. Each panel definition in a library is referred to as a library member. A library can be used as the sole source of panel definitions, or it can be concatenated with other libraries defined to the system. A concatenation of libraries is called a panel path.

Management Services supports multiple panel libraries. You can define as many panel libraries as you want. Path and library definitions can be created, altered, or deleted dynamically, without the need to restart the system. Panel libraries and definitions are maintained using MODS. See the *MODS Programming and Administration Guide* for more information.

A panel path is assigned to each user of Management Services. The path defines the libraries from which their panels are loaded. Whenever an NCL procedure issues the &PANEL verb for a specific panel, the system searches your path for that panel name. Each library within the path is searched in sequence until the correct panel is found.

### Procedure—Creating a Panel Library

Step 1. Define a VSAM dataset in which the panel libraries are to be stored.

You can use job S01LCALC in SOLVE.JCL as a model for this dataset.

Step 2. Allocate the VSAM dataset to Management Services.

This can be done by one of the following methods:

- Update the RUNSYSIN member to include a DD statement for the dataset—the region job or started task must then be restarted to pick up the extra DD statement
- Use the ALLOCATE command to dynamically allocate the dataset

Step 3. Open the VSAM dataset by using the UDBCTL command.

This command also initializes the dataset. For example:

```
UDBCTL OPEN=PROD ID=PROD
```

Step 4. Define the attributes and the file ID of the library by using the LIBRARY DEFINE command.

For example:

```
LIBRARY DEFINE PROD ID=PROD DESC=Production panels library
```

The library does not need to be open before the LIBRARY DEFINE command is issued, but must be opened before the library can be used.

Step 5. Define the paths of which the library is to be part.

For example, to define a path called EXAMPLE consisting of libraries PROD and PANELS concatenated in that order, use the LIBPATH DEFINE command as follows:

```
LIBPATH DEFINE EXAMPLE LIB=(PROD,PANELS) +
        DESC=Example panel path
```

Once the libraries and paths have been defined, you can specify them in the user ID definitions.

If no libraries or paths are defined in the INIT procedure, Management Services opens and defines the default panel library called PANELS. When necessary, this default configuration allows a system to ignore any concatenation processing, and to act as if it contained only the PANELS dataset.

See the *Management Services Command Reference* for a full description of the ALLOCATE, LIBRARY DEFINE, LIBPATH DEFINE, and UDBCTL commands.

# 4

# Implementing Additional Security

This chapter describes how to implement the various security exits that can be used with Management Services to provide additional security for your systems.

**This chapter contains the following topics:**

- About Implementing Additional Security
- Implementing an External Security Manager for User Definition
- Implementing Security for File Access
- Implementing INMC Link Security
- Implementing Dataset Allocation Authority
- Implementing Security for Dataset Services

## About Implementing Additional Security

Security exits allow you to interface with an external security system. This provides additional security to that provided by UAMS. Management Services supports security exits for user logon, links between systems, and access to datasets. The following five security exits are supported by Management Services:

- User ID security exit
- INMC security exit
- NCL authorization exit
- Dataset access authorization exit
- Dataset services authorization exit

The following sections provide details about the functionality that these exits provide and their implementation.

## Implementing an External Security Manager for User Definition

You can use an external security package to replace the user definition functions carried out by UAMS. Do this by using one of the following types of exit:

- Partial security exit—performs password validation and optionally controls user ID definitions. Management Services maintains user ID definitions on the UAMS dataset.
- Full security exit—performs all user ID definition and password functions. The UAMS dataset is not used.

### Functions Performed by the Exit

The following security functions can be implemented by using the user ID security exit:

- Control access to your system
- Perform logon verification
- Allow users to change their password
- Retrieve user ID definitions
- Update user ID definitions
- Add additional security exit functions
- List user ID definitions
- Add user ID definitions
- Delete user ID definitions

Each of these functions is described below.

For details on how to write the security exit and the parameter lists associated with each function, see Appendix F, *User ID Security Exit Support*.

## Controlling Access to Your System

Access to your system is controlled by providing user logon security. Your exit needs to be able to perform the following functionality:

- Verify whether the user ID is authorized to access the system
- Confirm the attributes and privileges that the user has when logged on to the system

The exit is called to accept logon attempts from the following sources:

- Native terminal logons
- TSO/TSS interface logons from the External Interface Package (EIP)
- ROF logons
- Operating system console logons
- System environment logons
- APPC user region logons
- Model user ID logons

When a user attempts to logon, the user ID and password (if applicable) is passed to the exit for confirmation that the logon can proceed.

When a partial security exit is installed, and the user ID is defined on the UAMS dataset, the user ID attributes are also passed to the exit.

### EIP Logons

EIP logons can originate from TSO or BCI. These calls do not need to supply a password.

**Note**

A VM/SP system can receive a TSO interface logon from an OS/VS system, so the exit should be written to handle this.

## ROF Logons

ROF logon requests might originate from domains that implement differing levels of security. In order to assist the exit to reach a decision with ROF logon requests, the following information is provided to the exit:

- The INMC link name of the domain from which the request came
- The domain ID (if available) of the originating domain
- A flag byte indicating whether the domain ID is present
- A flag byte indicating whether the ROF request originated from a domain that is different from the one identified by the INMC link name

The SIGNON command allows a password to be specified. The exit can check whether a password was specified for a ROF logon, and refuse the logon if no password was specified.

## Operating System Console Logon

When an operating system console is first attached to Management Services, a logon request is made. The exit can return user ID information applicable to the profile required for the particular console or it can indicate that the user ID is unknown. See Chapter 3, *Implementing Security*, for more information about system console user IDs.

## System Environment Logons

A logon call is generated for each system environment during system or procedure initialization. No passwords are associated with system environment logons.

If the exit rejects the logon by setting return code 24, Management Services assigns default values for the system environment logon. See Chapter 3, *Implementing Security*, for further details about system environment logons.

## APPC User Region Logon

When an APPC transaction is defined with conversation level security, an APPC user region logon is performed to validate and sign on the partner transaction program region.

Depending on the type of APPC logon being performed, a password may or may not be provided. If one is provided, it should be checked. If none is provided, it indicates that the region is being started from a known, valid environment. In this case the user ID should be validated, with no password check.

## User ID Modeling

Using a partial security exit allows more flexibility with model user IDs. The following scenarios can be specified in the exit:

- The SYSPARMS MODLUSER command can provide a system default model name that is supplied to the exit to create a new user ID.
- The SYSPARMS MODLUSER model user ID can be used, or an alternative model can be nominated to define a new user ID.
- A model user ID can be nominated to override the existing attributes of an already defined user ID.
- The exit can modify any or all of the individual attributes of that user ID by supplying a group of structured fields.

In this manner, the exit has complete control over both known and unknown users wishing to log on to Management Services.

When a model is specified, the exit changes the value of the 8-character user ID name addressed by word 3 of the Logon Request parameter list, to the user ID name for the nominated model user ID. You can define many model user IDs, for example, ADMIN, NETOP, SYSOP, or SYSPROG.

The exit should validate both user ID and password before allowing an automatically modeled user creation. Otherwise simple mistakes from mistyped user IDs might generate spurious user ID definitions on the UAMS dataset.

**Note**

Model users take precedence over changes to the user ID. If the exit specifies a model user ID but the model is itself not defined on UAMS the logon attempt is rejected with a user not known condition. This is the case regardless of whether the user is defined on UAMS or not, and the logon fails.

### *Password Status of Modeled User IDs*

By default, all users defined using the model user ID are new users, and all new users have to change their password when they first log on. This is not convenient if users have the same password for all systems.

To avoid this situation, the partial security exit must indicate that the logon password is correct and that the user ID is to be created and treated as an existing user ID. This means that a password change is not enforced before allowing the logon.

If an unknown user logs on and is automatically given a new user ID based on a model designated by the exit, the new user ID is created on UAMS.

| If... | Then... |
|---|---|
| The exit sets return code 0 | The new user ID is created but the user is not asked to change their password. |
| The exit sets return code 4 | The user is requested to change their password since the exit is indicating the password for that user ID has expired. |
| The exit sets return code 8 | The new user ID is created but the user is prompted to change their password since Management Services has classified them as a new user. |

## Allowing Logon Verification

Once a user is logged on to the system, there are times when their password needs to be verified. Password verification is needed in the following circumstances:

- When a user enters their password to resume use of a locked terminal
- When a user attempts to alter the MAI-FS details that utilize the &USERPW function
- For any NCL procedure using &SECCALL CHECK

This functionality is supplied by the &SECCALL CHECK verb. When there is no external security package being used, &SECCALL CHECK is handled by UAMS.

### Using Model User IDs

The following points should be considered when model user IDs are subject to logon verification:

- The coding of user logon and logon verification calls should be consistent in the logic used, including their dealing with model user IDs
- If a user ID is not defined to UAMS then the exit can be coded to either reject the user ID or specify a user ID to be used as a model
- If the SYSPARMS MODLUSR operand is specified, all calls that pass the user ID to the exit also pass the model user ID

### APPC Link Verification

Verification is also required when an APPC link is started and the link is defined with PASSWORD=EXIT.

The exit is designed to use SAF APPCLU class. When a request for an APPC link is made, session partners are passed to the exit as an entity defined to the SAF security system in the following form:

```
NETID.LU1.LU2
```

where LU1 is the requesting system and LU2 is the target system. To remove the necessity to define NETID.LU1.LU2 and NETID.LU2.LU1 with the same session key, you should specify the security exit to swap LU1 and LU2.

## Allowing Users to Change Their Password

To ensure the security of your system, users need to have the ability to change their password in the following circumstances:

- At any time by using the MODS PASSWORD command
- When they log on and their password has expired

## Allowing User ID Information to be Retrieved

When a user is logged on to your system, they can enter commands that require the retrieval of their user ID attributes. You need to provide this functionality in the following circumstances:

- When a SHOW OCS command is issued—name and location is obtained for all users currently using OCS
- When a PROFILE INITCMD command is issued— retrieves a copy of the user ID definition to update their OCS profile
- When an FTS TRANSMIT command is issued—verifies the user ID's FTS privileges
- For any NCL procedure using &SECCALL GET
- When a user ID definition is updated

The required user ID information is returned to Management Services as a set of structured fields.

If you have a partial security system, Management Services retrieves the definition of the required user ID from UAMS and presents its definition to the exit as a set of structured fields for inspection or modification before completing the request.

If you have a full security system, the security exit must provide all the structured fields for the user definition.

See the section, *Accessing User ID Attributes*, on page 4-11, for details on how the exit processes structured fields.

## Allowing Updates of User IDs

User IDs need to be updated when information pertaining to the user has changed. You need to provide this functionality in the following circumstances:

- When a user ID requires a change in its privileges
- When a user needs to update their user details
- For any NCL procedure using &SECCALL UPDATE
- For any NCL procedure using &SECCALL CHANGE with the FIELDS or DETAILS operands

Updating a user ID requires the ability to retrieve user ID information.

If you do not want to override any user attributes from UAMS, then the parameter list should be returned unchanged and the return code set to zero. This applies to partial security exits only.

The exit does not need to support the ability to update user IDs, but if it does not, some other method of changing a user's INITCMD and user details must be available.

## Adding Additional Security Functions

You can add your own functions to the security exit. For example, to obtain statistics about the exit's performance. The &SECCALL EXIT statement provides this functionality by allowing you to communicate between the security exit and NCL procedures. Communication is performed by passing the contents of nominated variables to your security exit.

On return to the NCL procedure the contents of the variables, passed as parameter areas, are placed in individual NCL variables named &1, &2, &3, and so on. A parameter area that was assigned a zero data length by the exit sets a null value.

**Note**

The exit can return only as many variables to the NCL procedure as were nominated on the original &SECCALL EXIT statement.

*Example*

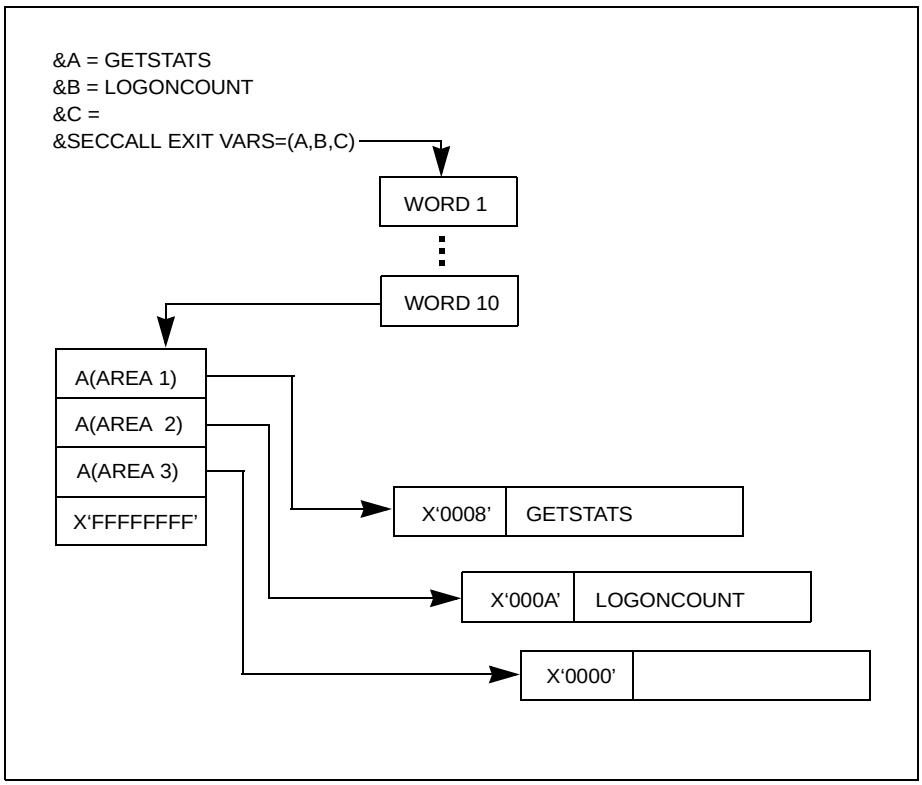
To obtain statistics about the number of logons performed by the security exit set the following variables:

```
&A=GETSTATS
&B=LOGONCOUNT
&C=
```

These variables are passed to the exit as parameters by the &SECCALL EXIT VARS=(A,B,C) statement. The exit modifies the contents of these parameters, setting the data length in the first 2 bytes of the parameter area. The maximum length of the parameter area is 256 bytes.

Figure 4-1 shows how the exit deals with an &SECCALL EXIT call.

*Figure 4-1. Representation of &SECCALL Exit Variables to Security Exit*




When the exit returns the variables in Figure 4-1 to the NCL procedure, the values of the parameter areas are placed in &1, &2, and &3.

The exit can also set a return code that is passed back to the NCL procedure as the value of the system variable &RETCODE. In addition, the exit can supply a message which is returned to the procedure in &SYSMSG.

&SECCALL EXIT is available for use in any NCL procedure, but is valid only in systems which are configured with a security exit. For detailed information about &SECCALL EXIT, see the *Network Control Language Reference*.

## Allowing Listing of User IDs—Full Security Exit Only

You can obtain a list of all the user IDs defined to your system. To build this list you need to retrieve sequential user IDs. This functionality is performed by the OPT=KGT or OPT=KLT option of the &SECCALL GET NCL statement. This statement retrieves user ID definition details for an NCL procedure of the next (KGT) or previous (KLT) user ID defined following a nominated user ID name.

If you use a partial security exit, this call is not made to the exit. A sequential get is performed on the UAMS dataset and then a specific return user ID information call is passed to the security exit.

## Allowing User IDs to Be Added

You need to be able to add a new user ID definition so that a new user can access the system. This functionality is provided by the following:

- The UAMS Add function
- Any NCL procedure using &SECCALL ADD USERID

If you use a partial security exit, and you do not want to override any of the attributes, the return code must be set to zero and the parameter list returned unchanged for the user definition to be added to the UAMS dataset.

If you use a full security exit, this function need not be implemented as it is performed by your external security package.

## Allowing User IDs to Be Deleted

You need to be able to delete a user ID definition. This functionality is provided by the following:

- The UAMS Delete function
- Any NCL procedure using &SECCALL DELETE USERID

If you use a full security exit, this function need not be implemented as it is performed by your external security package.

### Accessing User ID Attributes

If the user logging on is defined to UAMS, their user ID definition privileges are presented to the exit as a list of structured fields. This list is addressed by word ten of the logon request parameter list.

By translating the definition of a known user ID into structured fields and presenting them to the exit at logon time, the exit is given the opportunity to inspect or modify the attributes and privileges of the user who wishes to log on.

The exit cannot add structured fields to the list provided on the call but the following actions can be taken:

- Any structured field passed to the exit can be modified
- A structured field can be logically deleted from the list by clearing its address pointer from the list
- A complete replacement set of structured fields can be provided by the exit by replacing the address pointer in word ten of the parameter list.

See Appendix G, *Structured Fields*, for detailed information about the structured fields provided by Management Services.

## Implementing Security for File Access

The NCL authorization exit controls access to the following:

- User databases (UDBs)
- SQL databases in a DB2 environment
- Writing SMF records (OS/VS systems)

Access can be restricted to certain levels, or can be denied altogether. For example, the level of access to a database can be used to restrict users to read only or update without delete.

The exit is invoked automatically the first time an NCL procedure attempts to open a database, using the &FILE OPEN statement, or write a record using the &SMFWRITE statement. The level of access permitted is set by a return code from the exit and is available to the NCL procedure in the &FILERC system variable.

**Note**

If an exit controls the writing of SMF records in OS/VS systems, it needs to provide clearance for a user to use the &SMFWRITE verb.

Subsequent references to the same database by the same procedure do not invoke the exit again, unless the file connection is released by an &FILE CLOSE statement.

You can also define other uses for the NCL exit by tailoring the supplied NCLEX01 or writing your own.

## Activating the NCL Authorization Exit

To activate the NCL authorization exit, specify the name of the load module to be invoked in the SYSPARMS NCLEX01 operand. To ensure that the exit is always activated during system initialization, place the SYSPARMS command in the INIT initialization member. If no exit is required specify SYSPARMS NCLEX01=NO. If necessary the name of the exit can be changed on-line and a new module invoked.

For details of the SYSPARM NCLEX01 operand, see Appendix D, *SYSPARMS Operands*.

**Note**

If you are using a security product, such as RACF, to control VSAM file access, ensure that Management Services itself is authorized for file access.

The exit executes within a Management Services subtask and can therefore issue WAITs or SVCs that cause suspension of the task without affecting the Management Services primary task.

The exit is passed a parameter list that provides details of the request. Write the exit as a reentrant facility so that multiple concurrent requests are possible.

For performance reasons, calls to NCLEX01 are not generated to the high-usage system files MODS and $PSPOOL.

### Errors in the Exit

If an ABEND occurs within the exit, any current request is rejected as though access had been denied. Although this can impact the requesting NCL procedure, other processing will not be affected. Subsequent requests to the exit continue without impact.

### Parameters Passed to the Exit

When the exit is invoked, it is passed a communication area that provides details of the request and the requestor. The $NMNCEX1 macro, as supplied in the Management Services Macro library, describes this area. The macro is distributed as follows:

- OS/VS systems—NMMACLIB file
- VM systems—SOLVE MACLIB on the V3UTIL minidisk

For details of this communication area, see Appendix K, *The NCL Authorization Exit Communication Area*.

## Pre-loading the NCL Authorization Exit

You can load a single copy of the exit into the Management Services region in advance to avoid the overhead of the loading process. To do this, specify the name of the exit in the SYSPARMS MODLOAD operand.

If the exit is link edited with the RENT (reentrant) attribute this single copy is shared concurrently among multiple requestors.

**Warning**

It is strongly recommended that you make the NCL exit module reentrant. Failure to do so causes significant additional disk activity and resource consumption in a busy system, and can impact overall performance.

## Providing Additional Checking in the NCL Authorization Exit

You can provide additional checking for access through the NCL authorization exit. This additional information is specified on the &FILE OPEN statement.

*Example*

To have the exit ask for the user's password to confirm access to a particular UDB, the password information should be coded in the exit as follows:

```
&FILE OPEN ID=MYFILE DATA=&MYPASSWD
&FILE OPEN ID=MYFILE FORMAT=UNMAPPED DATA=&MYPASSWD
```

The exit will interrogate the password and set an appropriate return code to control the subsequent processing in the NCL procedure.

A maximum of 50 characters of data can be passed. This data is available to the exit in the NEXUDATA field. The length of the data passed is set in the NEXUDLEN field. No validation is performed on passed data.

## Correlating Authorization with Security Exit Authorization

If you have a partial or full security exit you can correlate access to UDBs by sharing information between your external security exit and the NCL authorization exit.

The NEXCORR field of the communications area in the NCL authorization exit is the standard correlator that can be used by your external security exit. This could, for example, be used to address a control block that contains information associated with that user.

If your external security exit supports a SAF user token, then this is also available in the NEXUTOKN field in the communications area. This user token is the UTOKEN provided by the external security exit when the user logs on to Management Services, and can be used in a security authorization call to verify the user's access to the UDB.

## Specifying UDB Authorization

The exit can set a variety of return codes which limit processing of a UDB by an NCL procedure. The return code is set in Register 15 on return from the exit. The following return codes can be set, and are available in the system variable &FILERC on completion of the &FILE OPEN statement. Values other than those described here result in an error and the procedure is terminated.

| Return Code | Meaning |
|---|---|
| 0 | Read only access |
| 4 | Update access (including addition of new records) but without delete |
| 8 | Unlimited access |
| 12 | Abort processing, no access permitted |

The $NMNCEX1 macro provides assembler language equates for the above return codes and it is advised that these be used. They are:

| Return Code | Assembler EQU |
|---|---|
| 0 | NEXR15RD |
| 4 | NEXR15UP |
| 8 | NEXR15DL |
| 12 | NEXR15AB |

**Note**

An additional return code, 16, is set as the value of &FILERC by NCL if the requested UDB is not available. This can occur due to the following reasons:

- The UDB does not have a logical name assigned to it
- The dataset fails to open during initialization or is closed by the UDBCTL command
- UDBCTL is used to stop the UDB

## Determining Access Levels from NCL Procedures

On return from the exit, the NCL procedure resumes processing with the exit return code available in the &FILERC system variable. The NCL procedure can test this system variable and process accordingly. For example:

```
FILE OPEN ID=MYFILE
&GOTO .&FILERC

.0 &WRITE READ ONLY ACCESS AVAILABLE.
                   :
.4 &WRITE UPDATE AVAILABLE WITHOUT DELETE.
                   :
.8 &WRITE UNLIMITED ACCESS.
                   :
.12 &WRITE ACCESS DENIED BY INSTALLATION EXIT.
                   :
.16 &WRITE FILE NOT CURRENTLY AVAILABLE.
                   :
```

**Note**

If the exit imposes any access limitations and the NCL procedure attempts to process outside of those restrictions, the NCL procedure will be terminated with an appropriate error message.

## Specifying Authorization to Write SMF Records

An NCL process that issues the &SMFWRITE statement to attempt to write a record to the MVS SMF dataset causes the authorization exit to be invoked to get permission for the SMF record to be written.

Permission is obtained once only for any process, regardless of the number of procedures that it invokes.

The $NMNCEX1 macro, which maps the parameter list passed to the exit, contains details of the information passed to the exit when an &SMFWRITE authorization call is made.

### Return Codes from &SMFWRITE Authorization Requests

The exit can return one of two return codes in response to a request to use &SMFWRITE. The return code is made available to the requesting procedure in the &FILERC system variable as follows:

| Return Code | Meaning |
|---|---|
| 0 | &SMFWRITE authorized and has been processed |
| 4 | &SMFWRITE not authorized. Record not written |

### Default SMF Authorization

By default, system level NCL processes are granted authorization to write SMF records using &SMFWRITE if no exit has been implemented. System level procedures include those executed by virtual user environments such as LOGPROC, PPOPROC, EASINET, and so on.

Processes invoked by private users, for example from OCS, are not by default allowed to use the &SMFWRITE verb. If you want to allow a private user to write SMF records you must implement an authorization exit to allow the request.

## Designing Your Exit

You should consider the following when planning your exit:

- If you code an NCLEX01 routine, it must always handle authorization for &FILE OPEN and &SMFWRITE.
- You should design the exit to reference a table of UDB names or DD names which require authorization. If a particular UDB is not in the table then further validation can be bypassed. This ensures that new UDBs can be introduced to the system and initially tested without the need to update the exit.
- The exit might need to handle other requests, such as &NDBOPEN, or &SQL.

- You should allow for undefined or unrecognized function codes, to accommodate future additions to these codes. It is suggested that unknown functions be acknowledged with a return code of 0.
- Consider whether you want to interface the exit with an external security system.

### Sample Distributed NCL Authorization Exit

The NCLEX01 member of the distributed installation library is supplied as a sample exit. Without modification this sample authorizes all requests. This distributed sample is reentrant and is therefore suitable for loading in advance using the SYSPARMS MODLOAD operand.

## Implementing INMC Link Security

A security exit is provided as one of the facilities of INMC (Inter-Management Services Connection). This provides security for the connections between domains.

The security exit is coded as an assembler language exit routine. An exit must exist in each of the domains that are being linked. The exits are installed with the following two components:

- Primary exit—communicates with the secondary exit in the other domain and decides whether to establish the link and allow traffic flow
- Secondary exit—responds to messages received from the primary exit and makes no decisions whether a link is established or terminated

This allows each end of an INMC link to determine independently whether a link should be activated, without having to depend on cooperation from the remote domain in order to enforce the decision.

The two components are described in more detail below.

## The Primary Exit

The INMC *primary* exit decides whether a newly opened INMC link to a remote domain should be made available for general traffic. The secondary exit is not involved in this decision.

In order to pass control to the exit to make this decision the following calls are made by INMC to the exit:

- The Initialization Call—made when an INMC link to another domain is activated. The primary exit is notified of the event and whether the remote domain is configured with a secondary exit
- The Deliver Call—made to the primary exit when INMC receives the reply from the (remote) secondary exit to a message sent to it earlier by the primary exit
- The Notify Call—made to the primary exit when INMC determines that an unexpected event (for example, link outage) has occurred which results in the deactivation of the link

For every piece of data sent by the primary exit, there is always a response from the secondary exit. This response can be one of the following:

- Data exchange is complete
- Link outage has occurred and no reply is possible

If any protocol errors occur, a hang or stalemate condition might occur between the primary and secondary exits. Care, therefore, must be taken when designing the flow of information between the exit pairs and the rules defined for the various parameter lists passed to the exits must be adhered to.

For details on how to code the INMC security exit and the parameter lists associated with each call, see Appendix J, *INMC Security Exit Support*.

## The Secondary Exit

The function of the secondary exit is to act solely as a respondent to any messages received from the primary exit in the remote domain. The secondary exit has no power to recommend activation or closure of the link.

Calls made to the secondary exit are as follows:

- The Initialization Call—made when an INMC link becomes active. The exit can then perform any initialization required before returning to INMC.
- The Deliver Call—made to the secondary exit to deliver a message from the primary exit. The secondary exit must respond with a reply to the message, even if that reply is a null message.

- The Disconnect Call—made if the primary exit decides that the conversation is to be ended or when a link outage occurs.

  This call allows the secondary exit to end cleanly and tidy up any allocated resources.

For details on how to code the INMC security exit and the parameter lists associated with each call, see Appendix J, *INMC Security Exit Support*.

## Implementing Dataset Allocation Authority

The Dataset Access Authorization Exit (NMDSNCHK) allows you to check whether a user attempting to dynamically allocate a dataset to Management Services is permitted to do so. This exit is invoked in two situations:

- When an ALLOCATE command is issued
- During FTS transmission processing

The exit exists as a load module (OS/VS and VM/SP) NMDSNCHK. This module must be accessible to Management Services. If the module is not found, no allocations can take place. The installation may code NMDSNCHK to perform any required security checking.

For more information on the use of NMDSNCHK with File Transmission Services, see the *FTS Planning and Installation* guide.

### Sample Distributed Exit

A version of NMDSNCHK is supplied in the Management Services distribution libraries. It comes in both load module (OS/VS and VM/SP) format, and in source form. The distributed version authorizes all requests. The source contains extensive documentation and can be used as the base product for tailoring to your installation's requirements.

For further details on writing and installing the dataset access authorization exit, see Appendix J, *INMC Security Exit Support*.

## Implementing Security for Dataset Services

The Dataset Services authorization exit, NMDSSCHK, is called periodically during Dataset Services processing to perform security related functions. You can use this exit to implement security procedures at the user level rather than at the Management Services level.

The NMDSSCHK exit is similar to the NMDSNCHK exit but has more specific calls and is only used by Dataset Services functions. NMDSNCHK is not used by the Management Services ALLOC/UNALLOC commands. To make use of the exit, it must be called NMDSSCHK and exist in the load library that contains your exit procedures.

For details of the Dataset Services interface ($DSCALL) see the *Network Control Language Reference* manual.

### Function Calls Made to the Exit

There are three main types of calls made to the exit:

- Exit Initialization call—performs any initialization required
- Logical Function call—represents a function that has been requested by the user. For example, a dataset allocation request or an OPEN request for a dataset.
- Exit Termination call—allows the exit to clean up any allocated storage areas and, if applicable, break the connection with any security subsystem that is being used

Check the DSSCFUNC field to determine the type of call being made. See Appendix I, *Dataset Authorization Exits Support*, for a detailed description of each call.

### Sample Distributed Exit

A sample version of NMDSSCHK is distributed with Management Services in the Management Services distributed libraries. This is in load module format and in source form. This distributed version authorizes all requests and contains extensive documentation. The sample exit is provided as a base for you to implement your own exit procedure.

For further details on writing the dataset services authorization exit, see Appendix I, *Dataset Authorization Exits Support*.

# 5

# Administering Signed-on Users

This chapter describes the administration of signed-on or disconnected users. The administration utilities have a particular benefit where multiple systems are connected by using INMC links.

**This chapter contains the following topics:**

- Introduction
- Using the Active User List Facility

## Introduction

All users must be defined to the Management Services security system by using one of the following:

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

For further information on security, see Chapter 3, *Implementing Security*.

At any time, only a selection of the defined users are actively signed-on to, or disconnected from, a given system. These user IDs can be displayed by using one of two methods:

- Using the SHOW USERS command in either the OCS window or the Command Entry function (see the *Management Services Command Reference*)
- Using the Active User List facility of Management Services (described in the following section)

## Using the Active User List Facility

This section describes the Active User List facility, which assists the help desk staff or systems administrator in performing the following functions for one or more domains connected by INMC links:

- Monitoring user activity
- Cancelling a user
- Disconnecting a user

These utilities are available from the Management Services : Primary Menu (shortcut /**MS** from any panel).

*Figure 5-1. Management Services : Primary Menu*

```
PROD---------------- Management Services : Primary Menu -------------------/MS
Select Option ===>

   U   - Userid Access Maintenance Subsystem                UAMS
   LU  - Active User List                                   USERS
   B   - Broadcast Services                                 BCAST
   D   - Managed Object Development Services                MODS
   S   - System Support Services                            SS
   I   - INMC Link Maintenance                              INMCL
   WS  - Data Warehouse Services                            WS
   E   - Help Messages and Error Codes                      CODES
   PS  - Print Services Manager                             PSM
   R   - Report Writer                                      REPORT
   LA  - Activity Log Administration                        LOADMIN
   L   - Activity Log                                       LOG
   X   - Exit

User ID ............... _________                            (Optional LU)
Link or Domain Name ....+____________                        (Optional LU L)


 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

To list active users, complete the following steps:

Step 1. Type **LU** in the Select Option ===> field.

Step 2. Type a value in the User ID field.

This field allows you to enter the leading characters of a user ID. 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.

Step 3. If required, type a value in the Link or Domain Name field.

This field allows you to identify the domain name(s) (INMC link names) that apply to your request. 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.

Step 4. Press ENTER—the System Support : User ID List is displayed (see Figure 5-2).

*Figure 5-2. System Support : User ID List*

```
PROD------------------- System Support : User ID List ------------------------
 Command ===>                                                  Scroll ===> PAGE

                   C=Cancel CF=ForceCancel D=Disconnect S=Show (MAI or Error)
    Link Name    UserID   Terminal
    (Local)      USER001  NMMAF054
    (Local)      USER007  NMMAF066
 ---------------------------------------------------------------------------
    PROD2        USER001  NMMAF066
 ---------------------------------------------------------------------------
    PROD3        SSUA1205 NO MATCHING USER(S) ON THIS DOMAIN
    **END**









  F1=Help      F2=Split     F3=Exit                 F5=Find      F6=Refresh
  F7=Backward  F8=Forward   F9=Swap
```

The System Support : User ID List allows you to identify which domain (within 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)

## Actions on the User ID List

For details of the actions available on the System Support : User ID List, press F1 (Help).

# 6

# Communicating Between Domains Using INMC

This chapter describes how to use INMC to communicate between domains. It covers the following topics:

**This chapter contains the following topics:**

- About Communicating Between Domains
- About INMC Links
- Controlling INMC Links
- Planning INMC Links
- Maintaining Link Definitions
- Providing Centralized Control of Connected Domains

## About Communicating Between Domains

The Inter-Management Services Connection (INMC) component of Management Services allows you to establish and operate links between domains.

### What Is a Domain?

Each instance of Management Services that is running is considered to be a domain.

The DOMAIN command is used to define the domain. The command includes:

- The name of the domain definition. This is the same as the name specified by the PRI JCL parameter
- The access methods that can be used to contact remote domains
- Access method specific details:
  - For VTAM the domain name is used as the VTAM ACB APPL name
  - For TCP/IP a domain definition that includes the IP name or the address and port number

For more information about the DOMAIN command, see the *Management Services Command Reference*.

### What Is INMC?

INMC provides a general-purpose data transport mechanism that allows a domain to communicate with one or more domains across a logical link known as an INMC link. An INMC Link can be either of the following:

- Simple
- Multipath

These two types of link are described below.

## Simple INMC Links

INMC provides basic functionality which is suitable for networks where there is only one physical network path between domains. Simple INMC provides the following functionality:

- It allows a domain to communicate with one or more other domains across a logical link.
- It allows a maximum of two sessions between any two domains. Each domain uses one session as the transmit leg of the link and the other as the receive leg.
- You can set up simple link definitions, which are stored in the VFS dataset, 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 domains. These are known as multipath links.

A multipath link allows a link between two domains to comprise up to sixteen communication sessions (that is, each domain opens up to eight sessions with the other domain). 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 dataset.

**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 used by Management Services 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 which use VTAM.

## Security

INMC provides a security exit, allowing verification of the identity of the remote domain.

## About 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)
- File Transmission Services (FTS)

### Traffic Flow on Simple Links

Data traffic flows between two domains across the twin-session link between the two domains.

Traffic is directed according to preference, across the session in which the transmitting domain is the primary end. Therefore, the link can be regarded as having a transmit leg and a receive leg for each domain. 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 *Planning INMC Links*, on page 6-13, for the types of links that can be defined.

INMC prioritizes the traffic so that ROF messages are scheduled with the highest priority and FTS the lowest. The multipath link definitions can be defined to control ROF and ISR traffic; however, FTS traffic is controlled as part of the FTS initiator definition.

### Controlling Links

Links are established by using the LINK START command.

Link definitions are retrieved from the VFS dataset.

Links are stopped by using the LINK STOP command, and can be reset by using the LINK RESET command.

Usage of these commands is described in subsequent sections of this chapter.

## About INMC Links

There are two types of INMC links:

- Static INMC links
- Dynamic INMC links

These are described in the following sections.

### Static INMC Links

Static INMC links are most useful between domains that regularly communicate with each other, and that always use the same domain names.

The link between two domains cannot be established unless each domain 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 domains to satisfy temporary connection requirements, without specifying permanent definitions for particular domains.

Dynamic links are established between pairs of domains. They occur when *both* of the following happen:

- One domain actively attempts to establish communications.
- The targeted domain is prepared to accept the incoming INMC request on a generic naming basis, without having been specifically instructed to communicate with the other domain.

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 domain without issuing any LINK START commands.

The DEFLINK command specifies two 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

## Establishing INMC Links

Use the LINK command to define domains to each other.

You can either 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 be used (VTAM, XNF, EPS, or TCPIP).
- The link name by which the remote domain is known by this domain, and by the operators of this domain. This can be up to 12 characters long and is intended to let you assign meaningful names to the various domains in your network. It might help if you always assign a link name with the same value as the SYSPARMS ID specified for the remote domain.
- The name of the domain definition which contains the access method details.
- A message prefix to be added to all messages received from the remote domain.
- The color and highlighting that is to apply to all messages received in OCS from the remote domain.
- The retry interval at which Management Services attempts to contact the remote domain following a link outage.

### Example

To establish a link with a remote domain in Chicago with a domain name of CHIC1472 and messages from the remote domain 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, then, to start communication between the two domains, you must also enter a LINK command definition to this domain from the CHIC1472 domain.

## Routing Commands

When the link is established you can route commands to the remote domain by using the link name.

### Example

To show the users of a remote domain 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 domains 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 domains attempt to re-establish communication automatically.

Once a dynamic link is established, it operates in exactly the same way as a static link. However, if contact is lost and the dynamic link is broken, then each domain acts differently:

- The domain that requested the link perceives it as a static link definition and automatically tries to reconnect.
- The domain 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.

For more information about starting INMC links, see the *Management Services User's Guide*. For more information about the LINK and DOMAIN commands, see the *Management Services Command Reference.*

## Controlling 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 domain attempt link establishment to other domains automatically.

### Starting an INMC Link

To start an INMC link, use the following command:

`LINK START=`*`linkname`*

where *linkname* is the name by which the remote domain is known. If no previous LINK START command has been issued for this destination, Management Services retrieves the link definition stored under the link name from the VFS database and attempts to open an INMC link to the remote domain. 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.

## Stopping an INMC Link

To stop an INMC link, use the following command:

`LINK STOP=`*`linkname`*

This command terminates all sessions with the remote domain 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.

## Resetting an INMC Link

To reset an INMC link, use the following command:

`LINK RESET=`*`linkname`*

This command is only valid 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.

## Displaying INMC Links

To display the current status of an active link to a remote domain, perform the following steps:

Step 1. Type **/INMC** at the Select Option ===> prompt on any panel and press ENTER.

The INMC : Link Status List panel is displayed. This lists the sessions linked to the remote domain.

*Figure 6-1. INMC : Link Status List Panel*

```
PROD----------------------- INMC : Link Status List ------------------------
Command ===>                                                  Scroll ===> CSR

                                                                   S/B=Browse
    Link Name    Domain   Msgid Mode Status
    $RM$DENM2    DENM2    $RM$  S    ACTIVE
    $RM$QANM71B  QANM71B  $RM$  S    ACTIVE
    DENM13       DENM13   DE13  R    ACTIVE
    DENM15D2     DENM15D2 DE15  R    PEND-ACT
    DENM5        DENM5    DE5N  S    PEND-ACT
    QANM11       QANM11   QANM  S    PEND-ACT
    STNM3        STNM3    ST3N  S    ACTIVE
    **END**









 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find     F6=Refresh
 F7=Backward  F8=Forward   F9=Swap
```

For a description of the fields shown on this panel, press F1=Help.

Step 2. To view further details of a link's status, type **S** or **B** beside the link name and press ENTER.

The INMC : Link Status Display panel is displayed. The information displayed on this panel varies slightly depending on such things as:

- Whether the link is defined and has a description
- Whether the link has ever been active

*Figure 6-2. INMC : Link Status Display Panel*

```
PROD---------------------- INMC : Link Status Display ------Columns 00001 00079
Command ===>                                                  Scroll ===> CSR

Link name  STNM3

Link Status ............. ACTIVE      Transmission Mode ....... SIMPLE
Domain .................. STNM3
Domain ID ............... ST3N        NM ID ................... STNM3
Job Name ................ STNM3       Management Services ..... V3.6
Operating System ........ OS/390      System Name ............ SDD1
VTAM SSCP Name .......... SDD1VTM1    Host PU Name ........... ISTPUS

MsgID ................... ST3N
Color ................... PINK        Highlight ............... NONE
Primary Sessions ........ 1           Secondary Sessions ...... 1
Transmit Buffers (Kb) ... 35          Receive Buffers (Kb) .... 7

 S Logmode  Status      Size Am     Access Method Details
                        Count       Session comments
 1          ACTIVE       17K VTAM   NET=FTI      CID=120000B0


                                   Last Active: 07.03.25 on 06-JUN-2001
 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find
 F7=Backward  F8=Forward   F9=Swap
```

For a description of the fields shown on this panel, press F1=Help.

## Improving 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)

### Increasing 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**

For more information about the INMCBFSZ operand, see Appendix D, *SYSPARMS Operands*.

## Increasing RU Size in Mode Table Definitions

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, then 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, then 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. See the following section, *Specifying a Particular Route for INMC Traffic*, for an example of a logmode table definition.

The minimum RU size allowed for INMC sessions is 1 kilobyte. If no RU sizes are specified, or if a value less than the minimum size is set, then Management Services uses a default value of 1 kilobyte.

## Specifying 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 allow 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 FTS. It ensures that data transfer does not compete with other traffic.

It is recommended that when defining multipath links, you associate each session with a separate logmode table entry, where each logmode entry has a separate COS definition that defines a single virtual route.

See the appropriate VTAM installation manuals for further details on logmode table definitions.

### Using Security with INMC Links

INMC provides a security exit which allows you to implement security to determine whether a link between two domains 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 domain and exchanges any identification information that might be required.

For details of the parameter lists and coding requirements for the INMC Security Exit, see Chapter 4, *Implementing Additional Security*, and Appendix J, *INMC Security Exit Support*.

### Diagnosing Problems with INMC Connections

If problems occur when establishing INMC sessions between domains, then a tracing facility is provided to help you determine problems. This is provided by setting the SYSPARMS LNKTRACE and SESSMSG operands. See Appendix D, *SYSPARMS Operands*, for more information about these SYSPARMS.

## Planning INMC Links

A link between two domains can comprise one or more communication sessions, termed a session group. Each domain can establish and operate up to eight sessions with the remote domain. The remote domain acts as the secondary end of each session.

The simplest link consists of two sessions between two domains, one in each direction. However, the link still functions successfully when only one session is open between the two domains.

There are three categories of link that can be established between domains:

- Simple-mode links
- Rotate and backup multipath links
- Preferential multipath links

Two domains 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 domain 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.

## Planning Simple-mode Links

A simple-mode link allows each domain 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, then the link is broken. This type of link is best suited to network configurations which provide a single path between two domains.

There are two ways in which a simple-mode link can be defined:

- 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 domain name of the remote domain
- The one- to four-character MSGID used as a prefix on ROF messages received from the remote domain

All other session parameters are allowed to default as dictated by the access method when sessions are established.

This way of defining domains to each other is appropriate in the following environments:

- There is a single physical network route providing the communication path between the two domains.
- There is no FTS traffic between the two CPUs.
- There is FTS traffic between the two domains, 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. See the section, *Specifying a Particular Route for INMC Traffic*, on page 6-12, for details on specifying COS definitions.

**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 domains 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.

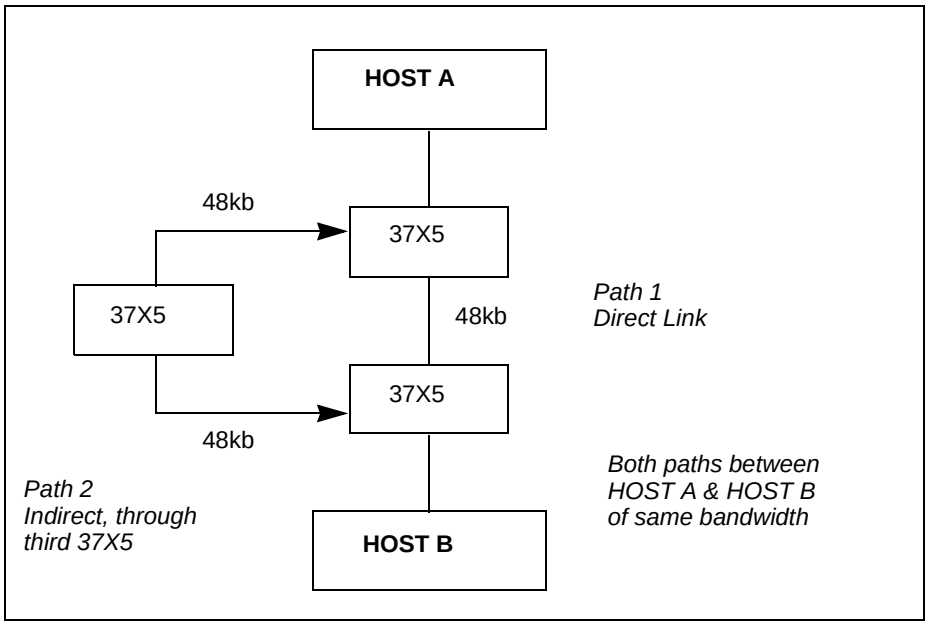
## Planning Rotate and Backup Mode Multipath Links

Rotate and backup mode links provide multiple paths through a network connecting two domains. This type of link can establish from two to eight sessions with the remote domain. 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 domains 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.

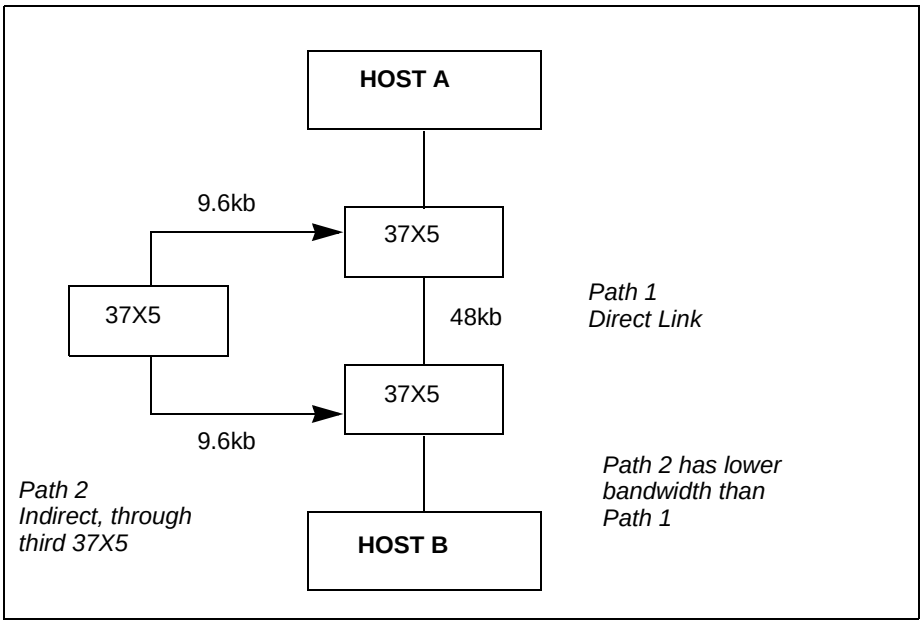
Rotate and backup links are appropriate in a network where there are two or more active physical network paths of similar bandwidth between two domains. A link is required that uses the aggregate bandwidth of all those physical paths. Figure 6-3 demonstrates such a configuration.

*Figure 6-3. Rotate Link Configuration*



Rotate and backup links are appropriate in a network where there are two active physical paths of different bandwidth between the two domains. A link is required that uses the high-speed path exclusively, but switches automatically to the slower path if the fast path fails. Figure 6-4 shows a representation of such a configuration.

*Figure 6-4. Backup Link 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.

## Planning Preferential Mode Multipath Links

A preferential link definition defines from two to eight sessions for a remote domain, 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 available 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 domains in the same order as it left the transmitting domain. This occurs regardless of network route bandwidths which could lead to faster transmission across one session than another.

Failure of one session within 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.

## Maintaining Link Definitions

Before you can establish multipath links, you need to specify a link definition in each domain that is going to communicate with another.

**Note**

You can also define simple mode links. This allows you to specify session parameters such as the logmode table entry name.

These link definitions are defined by using the INMC link definition facility of the System Support Services feature of Management Services. The three types of link that you can define are:

- Simple-mode link
- Rotate and backup
- Preferential

The link name is the symbolic name by which the remote domain is identified. You should choose a link name that is meaningful, so that specific domains can be readily identified by their link name. It is also recommended that each domain is known to all other domains by the same link name.

The link definitions are saved on the Management Services VFS database for retrieval when a link to a remote domain is to be activated.

Only those users authorized for System Support Services can define INMC link definitions.

## Displaying INMC Link Definitions

To display a list of INMC link definitions, enter **/INMCDEF** at the ===> prompt on any panel.

The INMC : Link Definition List panel is displayed, showing an alphabetic listing of all INMC link definitions on the VFS database.

*Figure 6-5. INMC : Link Definition List Panel*

```
PROD-------------------- INMC: Link Definition List --------------------------
Command ===>                                                  Scroll ===> CSR
                                        S/B=Browse U=Update C=Copy D=Delete
  Link Name   Domain   Msgid Mode Sess Comments
  $RMVEX      ANY      SSO   R       2 Supports VTAM and EPS access
  $RMVXX      ANY      SSO   S       1 Supports VTAM access only
  $RMXET      ANY      SSO   R       2 Supports EPS and TCPIP access
  $RMXEX      ANY      SSO   S       1 Supports EPS access method
  $RMXXT      ANY      SSO   S       1 Supports TCP/IP access method
  ACNMT22     ACNM122  ACN2  S       1
  AUDJB1NM    AUDJB1NM AUDJ  S       1
  LINK01      PROD4    LK01  S       1 This is a high speed link
  LINK02      PROD1    LK02  S       1
  LINK03      PROD2    LK03  S       1
  NMVT        NMVT     NMVT  P       4 High speed link to VM
  SDJSCV      ACNMA    TEST  S       1
  SD17        SD17     SD17  S       1 Link to sd17
  SD4         SD4      SD4   S       1
  TEST        SD5      SD5   S       1 Test link definition




 F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

For a description of the fields shown on this panel, press F1=Help.

From the INMC : Link Definition List panel you can browse, update, copy, or delete a current link definition. You can also create a new link definition.

## Creating 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.

### Creating a Simple-mode Link Definition

Step 1. From the INMC : Link Definition List panel, press F4=Add.

The INMC : Link Definition Details panel is displayed.

*Figure 6-6. INMC : Link Definition Details Panel*

```
PROD------------------ INMC : Link Definition Details -----------------------
Command ===>                                                  Function=ADD

Link Name ................. ____________

Link Definition Comments:   ____________________________________________
                            ____________________________________________


Remote domain ............. ________
Message ID ................ ____
Color ..................... _________    (Green, Red, Pink, Turquoise,
                                          Blue, Yellow, White, None)
Highlight ................. _______      (Reverse, Uscore, Blink, None)
Transmission Mode ......... ____________ (Simple, Rotate, Preferential)

For Simple Mode:
 Access Method ............ _____        (Vtam, Tcpip, Xnf, Eps)
 Logmode .................. ________
 Comments ................. ________________________________________


F1=Help      F2=Split     F3=File      F4=Save      F5=Sessions
                          F9=Swap                                F12=Cancel
```

Step 2. In the Destination Link Name field, type the link name (1 to 12 characters) that is to be added.

Step 3. Fill in each of the fields, using **S** in the Transmission Mode field, and press F3=File.

The definition is saved on the VFS database. The INMC : Link Definition List panel is redisplayed, accompanied by a message confirming successful addition of the definition.

## Creating a Rotate and Backup INMC Link Definition

Step 1. From the INMC : Link Definition List panel, press F4=Add.

The INMC : Link Definition Details panel is displayed.

Step 2. In the Destination Link Name field, type the link name (1 to 12 characters) that is to be added.

Step 3. Fill in each of the fields, using **R** for the transmission mode field, and press F5=Sessions.

The INMC : Rotate and Backup Session Details panel is displayed.

*Figure 6-7. INMC : Rotate and Backup Session Details Panel*

```
PROD--------------- INMC : Rotate and Backup Session Details ------------------
 Command ===>                                                Function=ADD


   Link Name:

 Sess  Logmode Status  Acs-meth  Session Related Comments
  1    ________  ___    _____    ________________________________________
  2    ________  ___    _____    ________________________________________
  3    ________  ___    _____    ________________________________________
  4    ________  ___    _____    ________________________________________
  5    ________  ___    _____    ________________________________________
  6    ________  ___    _____    ________________________________________
  7    ________  ___    _____    ________________________________________
  8    ________  ___    _____    ________________________________________

  ROF Preferred Session ........ _
  ISR Preferred Session ........ _



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

This panel allows you to specify details of the number of sessions to be opened, the logmode table entries associated with each of those sessions, and the session. For a description of the fields, press F1=Help.

Step 4. Fill in all of 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 is redisplayed, accompanied by a message confirming successful addition of the definition.

## Creating a Preferential INMC Link Definition

Step 1. From the INMC : Link Definition List panel, press F4=Add.

The INMC : Link Definition Details panel is displayed.

Step 2. In the Destination Link Name field, type the link name (1 to 12 characters) that is to be added.

Step 3. Fill in each of the fields, using **P** for the transmission mode field, and press F5=Sessions.

The INMC : Preferential Session Details panel is displayed.

*Figure 6-8. INMC : Preferential Session Details Panel*

```
PROD---------------- INMC : Preferential Session Details ----------------------
 Command ===>                                                Function=ADD


   Link Name:

 Sess  Logmode  Count  Acs-meth  Session Related Comments
  1    ________ ___    _____    ________________________________________
  2    ________ ___    _____    ________________________________________
  3    ________ ___    _____    ________________________________________
  4    ________ ___    _____    ________________________________________
  5    ________ ___    _____    ________________________________________
  6    ________ ___    _____    ________________________________________
  7    ________ ___    _____    ________________________________________
  8    ________ ___    _____    ________________________________________

  ROF Preferred Session ........ _
  ISR Preferred Session ........ _




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

This panel allows you to specify the number of sessions to be opened, 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 these fields, press F1=Help.

Step 4. Fill in all of 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 is redisplayed, accompanied by a message confirming successful addition of the definition.

## Modifying an INMC Link Definition

After an INMC link definition has been created, it can be updated at any time. To modify any of the values specified in a link definition, perform the following steps:

Step 1. On the INMC : Link Definition List panel, type **U** beside the name of the link that you want to update, and press ENTER.

The same panels that were used to add a new definition are displayed with the values that were originally entered.

Step 2. Change the required values on the panels and press F3=File to save the changes.

The INMC : Link Definition List panel is redisplayed, accompanied by a message confirming successful update of the definition.

## Deleting an INMC Link Definition

To delete a link definition, perform the following steps:

Step 1. On the INMC : Link Definition List panel, type **D** beside the name of the link that you want to delete, and press ENTER.

Step 2. Press ENTER again to confirm the delete.

The link definition is deleted.

## Providing Centralized Control of Connected Domains

There are two features of Management Services which provide services for centralized control of domains connected using INMC:

- Remote Operator Facility (ROF)—centralized control at the *user* level
- Inter-System Routing (ISR)—centralized control at the *system* level

### Using ROF for Centralized Control of Remote Domains

ROF provides you with the ability to monitor and control remote domains from the local domain through OCS. ROF uses INMC as its transport facility. ROF services are only available between two domains when there is an active INMC link between them.

For information on how to operate ROF sessions, see the *Management Services User's Guide*.

#### Defining 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 domains?
- 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 domains to which it can log on?
- Do the domains have a separate or shared UAMS dataset?

##### *Displaying ROF Users*

To identify which users are operating ROF sessions, either to or from remote domains, use the SHOW USERS command.

#### Including 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. These simplified commands can be included in the system initialization procedures RMINIT or RMREADY. These EQUATE commands are then available as system-wide defaults.

See the *Management Services Command Reference* for details about the EQUATE command.

### Using ROF with NCL Processes

Any NCL procedure started by a ROF operator can be used to issue commands to remote domains. 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 within an NCL procedure to issue commands to remote domains. The results from these commands are returned to the response queue of the issuing procedure and can be read using &INTREAD.

### Using ROF from Background Environments

Background environments can create ROF sessions with remote domains. 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 domain then the sign on fails.

Background environments can route commands to a remote domain over a ROF session and have the results returned to it for examination.

### Using ROF from the System Console

System Consoles can establish ROF sessions with remote domains 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 domain if a ROF session is to be established.

If you are using the default console user ID (.DFLTOP), then ROF sessions are not supported.

## Using ISR for Centralized Control of Remote Domains

Inter System Routing (ISR) is used to send data from one domain to other domains 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 domains

The following system environments can use ISR to forward the data they receive to other domains:

- AOMPROC—receives system console messages and related information
- PPOPROC—receives unsolicited VTAM messages
- CNMPROC—receives CNM data from VTAM
- NTS—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 domain.

## Planning for ISR

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 domains
- In which domains 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 via ISR, the following rules apply:

- ISR does not return data to the domain from which it was sourced
- ISR does not return data to the domain from which it last came

This protection is applied to avoid the possibility of one or more messages being continually sent from domain to domain in a message loop.

However, in some configurations, a message, sourced from outside a group of connected domains, 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.

## Establishing ISR Conversations

To establish an ISR conversation, use the ISR command. The ISR command allows you to enable, disable, or alter a conversation. It also allows you to control the type of data that is transferred across an INMC link. Each conversation class uses its own version of the ISR command. See the *Management Services Command Reference* for details on how to use the ISR command.

## Monitoring the Status of ISR

You can user the SHOW ISR command to monitor 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 three types of status are displayed:

- A local status—information about the ISR conversations from the perspective of the local domain
- A remote status—information about ISR conversations from the perspective of the remote domain
- An actual status—the status of the ISR conversations from the combined perspective of both the local and the remote domain

Figure 6-9 displays an example of SHOW ISR command output.

*Figure 6-9. 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 STNM3        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*
_____________________________________________________________________________




-------------------------------- NetMaster ---------------------------------
==> show isr
```

See the *Management Services Command Reference* for detailed information on the use of the SHOW ISR command.

## Controlling the Type of Data Received Through ISR

In the case of AOM, PPO, and NEWS, the user can control data exchange though use of AOMPROC, PPOPROC, and CNMPROC respectively. In the case of NTS, control is through the domain and class parameter settings that request the type of data NTS is to collect.

Full details on the interaction of each component with ISR can be found in your product implementation and administration guide.

However, in the case of AOM, PPO, and NEWS, there are important parallels that are discussed in the following section to provide a general understanding of ISR operation in these environments.

### *Common ISR Functionality*

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 data exchange between the environments is controlled by the ISR command. Internal or programmatic control is possible within a procedure by operands on the &*xxx*CONT and &*xxx*DEL NCL statements.

  **Note**

  For the purposes of this section, *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.

  See the *Network Control Language Reference* for detailed information about &*xxx*READ, &*xxx*CONT, &*xxx*DEL, or &*xxx*REPL.

- Delivery of data to other domains via ISR can be handled in two ways:
  - 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, NCL return code and feedback information indicates the result of the request.
  - Explicit delivery—specific destination operands are specified on the &*xxx*CONT and &*xxx*DEL statements. These are the LINK, DOMAIN, and SSCPNAME operands.

    **Note**

    Only one of these operands can be specified at any one time. The use of all of these operands might not be supported by all components.

    For example, if you want to specify that data received from AOMPROC is only sent to one of your two remote domains, you need to identify this domain on the &AOMCONT statement.

## Running ISR Data Flow Without a Procedure

If a procedure terminates for whatever reason, or no procedure is started, Management Services 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 domain.

# 7

# Using APPC to Communicate with Other Systems

This chapter describes how to use Advanced Program to Program Communications (APPC) to communicate between domains and other remote systems.

**This chapter contains the following topics:**


- About APPC
- Defining APPC Links
- Defining APPC Tables to Management Services
- Implementing Security for APPC Links
- Activating APPC Links
- APPC Link Definition Examples
- APPC Application Programming Facilities

## About APPC

APPC allows programs to communicate and exchange data using a common set of communication protocols. Communication takes place between programs using LU Type 6.2 sessions.

APPC is supported on a wide variety of both IBM and non-IBM platforms. This allows distributed applications to be developed in a heterogeneous networking environment. Although APPC standardizes communication between such applications, the programs themselves can reside on differing hardware and software platforms, and be written in different programming languages.

APPC employs a peer protocol. This means that a program's APPC behavior is not restricted in operation due to the network node where it resides. Nor is the node's communications ability a determining factor in application design.

APPC provides a set of common services available to an application through an architected verb set. The verb set has a direct relationship with underlying LU6.2 session protocols, but the application itself is written in a manner totally independent of SNA sessions. Only those internal services that support the verb set in a particular product implementation are concerned with managing session activity.

### Transaction Programs and Conversations

When two programs communicate using APPC, they do so via a communication path called a *conversation*. All activity that occurs as a result of APPC communication is called a *transaction* and the programs involved are known as *transaction programs*. Terms such as client/server processing or distributed transaction processing, are often used to describe this form of data processing.

One feature of APPC is that two programs do not have to be running in order to start communications. APPC allows a program in any LU to request the invocation of a program in any other LU in order to fulfil its processing requirement. The program achieves this through a process known as *allocation*, which establishes a conversation, and sends an *attach request* that the nominated program be invoked to service the remote LU of the conversation.

The invoked program is said to be *attached* in the remote LU. An attached program can itself attach other programs. These all form part of the same transaction, but each communication instance between programs is considered a separate conversation.

Within Management Services, any NCL procedure can issue an allocation request to start a conversation with another procedure or program. These can be in the same or another network LU. When Management Services receives an incoming request to attach a program, an NCL procedure is started to service the conversation.

## Conversations and Sessions

Each conversation is mapped to an LU6.2 session. While a conversation is active it has exclusive use of that session. An active conversation must terminate before any other can begin on that same session. This means that any one session is seen by LU6.2 conversations as a *serially-shared resource*.

### Concurrent Conversations

To conduct concurrent conversations between any two programs, the use of parallel sessions must be supported between them. The number of concurrent conversations that can be established between programs is dependent upon the number of parallel sessions available.

**Note**

Applications running under a PU Type 2.0 node, such as PC-based applications, must always play the role of Secondary LU and are restricted to a single LU6.2 conversation at any given time. Use of Node Type 2.1 and independent LUs allows such devices to play the role of a Primary LU (BIND sender) and also allows a parallel session capability.

### Sharing a Single Session

To share a single session for all transactions, conversations should be relatively short-lived. The conversation turnover and available session limit should be considered in program design and session configuration.

### Conversation Handling

Within Management Services, NCL procedures acting as transaction programs are concerned only with the management of their conversations. All aspects of conversation to session mapping and session management are handled by the LU Services components of Management Services and VTAM.

## Session Polarity

Each LU6.2 session has a designated polarity indicating which LU is the contention winner. This polarity can be negotiated during the session BIND phase but is then fixed for the session.

The LU designated as contention winner controls conversation allocation activity for the connection. It has the right to initiate conversations at any time without notifying the LU at the other end. Conversely, the LU designated as contention loser must request permission from the contention winner LU before it can start a conversation on that session.

For a parallel session connection the typical configuration is to support enough contention winner sessions in each direction to service concurrent demand for conversation initiation from each LU of the connection.

## Conversation Types

There are two types of conversation supported by APPC:

- Basic conversations—for use by SNA Services programs only
- Mapped conversations—support user transaction programs

Both basic and mapped conversations use the LU 6.2 session in the same way. The only difference is the manner in which they are operated at the LU6.2 protocol boundary. At this boundary there is one verb set for mapped conversations, and another for basic conversation. There are also some type-independent verbs.

Management Services supports the allocation of conversations as mapped or basic Because of the high level nature of the NCL verb interface all conversation operation is equivalent to the LU6.2 mapped conversation verb set.

## APPC Option Sets

APPC consists of basic facilities that all implementations must support. In addition to these facilities there are other more advanced facilities that are named option sets. An option set is comprised of a number of optional facilities. Optional facilities are grouped into sets so that the implementation choices are restricted to these sets, and not each individual option.

A full list of all LU6.2 Option Sets is contained in the *SNA Transaction Programmer's Reference Manual for LU Type 6.2*. The following section describes the option sets supported by Management Services.

### Management Services APPC Option Set Support

The following option sets are supported by Management Services:

**Conversations between programs located at the same LU**
Allows a program to allocate a conversation where the remote program is in the same LU as the local program.

**Delayed allocation of a session**
Allows a program to allocate a conversation and begin sending data before a session can be assigned to support the conversation.

**Immediate allocation of a session**
Allows a program to allocate a contention-winner session only if one is immediately available, otherwise the allocation fails.

**Session-level LU-LU verification**
Provides LU security by designating a password to verify the identity of a remote LU.

**User ID verification**
Allows a program to allocate a conversation with user verification by means of a supplied password or an already-verified indicator where appropriate.

**PIP data**
Allows a program to allocate a conversation and supply program initialization parameters to the attached remote program.

**Logging of data in a system log**
Allows a program to record error information in a system log.

**Flush the LU's send buffer**
Allows a program to explicitly cause data transmission on a session.

**Prepare to receive**
Allows a program to change from send to receive state with a number of different options.

**Receipt with wait**
Allows a program to operate a number of conversations and be notified when information is available on any one of them.

**Receive immediate**
Allows a program to request any data on a conversation but continue processing if none is available.

**Test for request-to-send received**
Allows a program to test whether a request to send indication has been received on a conversation.

**Data mapping**
Allows a program to request the mapping of data by the local and remote LUs.

**Get attributes**
Allows a program to obtain information about a mapped conversation.

### Supported APPC Products

The Management Services implementation of APPC allows you to communicate with the following products:

- Other domains
- CICS/OS/VS V1.7
- OS/2 Comms/Manager
- CommLink SNA/LU6.2
- MVS/APPC

It is possible that other APPC products that have not been tested will also operate with Management Services.

## Defining APPC Links

APPC links can be defined in two ways:

- Statically—using the LINK command
- Dynamically—using APPC table definitions

The type of APPC link you define is dependent on your needs. A static APPC link is used for one-off APPC connections. A dynamic APPC link is necessary if you want APPC links to be established on demand.

For information on how to use static APPC links, see the *Management Services User's Guide*.

### Defining Dynamic APPC Links

To define dynamic APPC links identify the components that can take part in APPC communication and their operational requirements to the local domain. These definitions are kept in APPC tables maintained by a set of Management Services (MS) commands. The table definitions perform the following functions:

- Relieve NCL from the task of fully specifying all the parameters required to initiate a conversation. This simplifies allocation requests.
- Provide validation for both locally and remotely initiated conversation requests. This provides a level of integrity before allowing communication to take place.
- Include information concerning destination attributes relevant to LU6.2 usage and for controlling resources such as sessions. This information allows Management Services to modify its behavior to suit a variety of LU6.2 implementations.

- Most importantly, these tables allow on-line maintenance of LU6.2 operational facilities, including aspects of control over NCL transactions without resorting to the modification of NCL procedures.

The APPC tables support the separation of the APPC programming facilities from the session support facilities. This allows NCL procedures to be developed and maintained in isolation from the changing network requirements.

There are four control tables that can be defined:

- Transaction Control Table (TCT)—associates transaction program identifiers with transaction program names
- Dynamic Link Table (DLT)—defines those LUs with which the local domain allows LU 6.2 sessions to be established
- Option Set Control Table (OSCT)—defines those options supported by a particular LU that are connected to Management Services through LU 6.2 session communication
- Mode Control Table (MCT)—defines mode names, the session characteristics they represent, and the number of sessions they support

## APPC Table Requirements

Although there are four APPC tables and each supports many parameters, it is not necessary to define all of the tables or parameters. In most cases the system provides default values. The following gives an indication of table definition requirements:

- Transactions for which both ends of the conversation run in the same domain, need only a TCT definition to define for each transaction the name of the NCL procedure that will service the request.
- LU6.2 connections that use the default option set, only need TCT and DLT definitions. This is sufficient to authorize single session connections to service conversations requiring no security and no data mapping support, and is appropriate for typical communication between Management Services and work stations running APPC.
- LU6.2 connections that use advanced options (for example, specific logmode) need all four table definitions. Such advanced options are likely to be required to enable sophisticated use of APPC between one domain and other domains or CICS systems.

## How APPC Control Tables Interact

An APPC link can be started from either the remote or the local domain as a result of a conversation allocation request (for example, an &APPC ALLOCATE) or an operator command (for example, LINK START). In either case the relevant DLT entry is located using the link name or LU name depending on which one was provided.

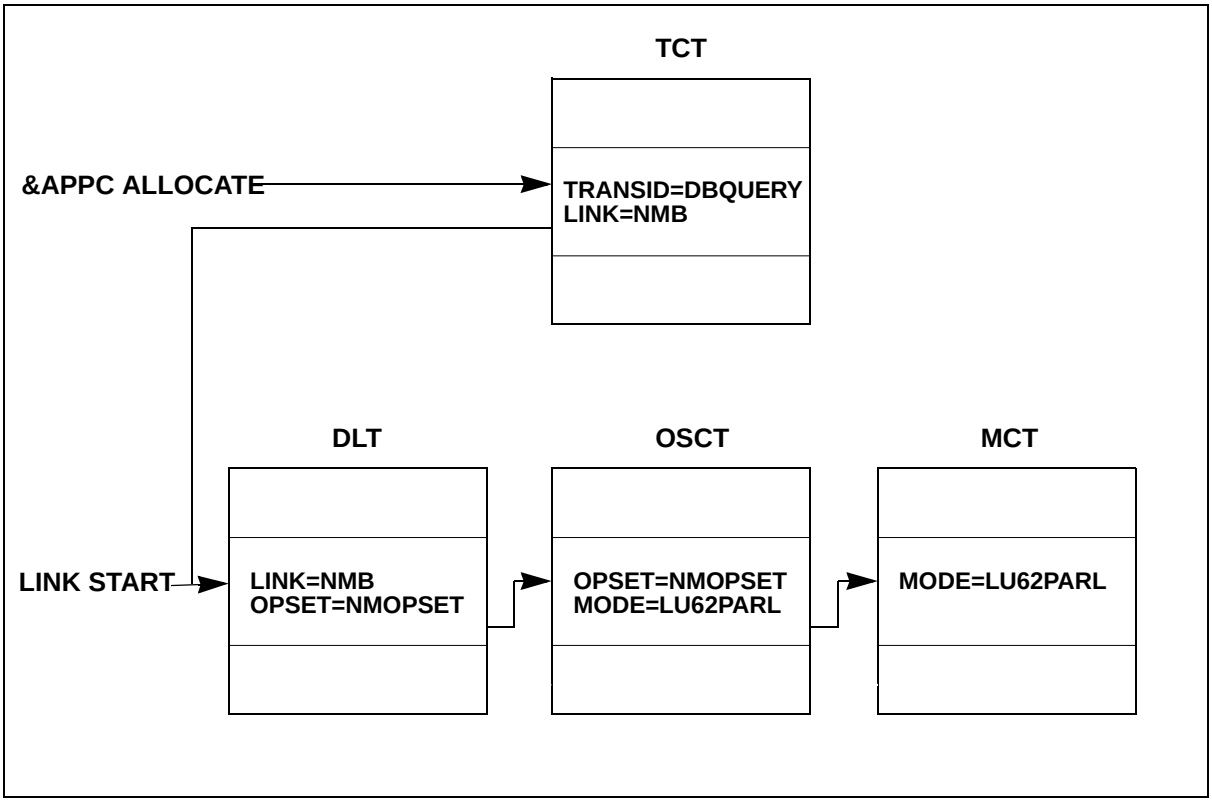
A DLT definition is needed for each remote LU that is to be connected to Management Services via LU6.2 sessions.

Link activation processing will then locate the OSCT entry nominated in the DLT definition and its associated MCT entries. The information from these tables is used to activate a link with the desired properties (for example, parallel sessions). If no OSCT or MCT entries are associated with a DLT definition default options and modes are used.

*Example*

When an NCL procedure issues an allocation request the transaction ID is used to locate the TCT entry. If no destination information was provided in the &APPC ALLOCATE verb the link name or LU name in the TCT are used (if defined) to determine if the link is active. If the link is not active the LU name or link name is used to locate the relevant DLT entry and link activation proceeds as described in Figure 7-1.

*Figure 7-1. APPC Table Interactions During Link Activation*




## Defining APPC Tables to Management Services

The following sections describe how to define each of the APPC control tables.

### Defining a Transaction Control Table (TCT)

The TCT is used to define all transactions to Management Services. Each TCT entry can be used to control allocation requests, attach requests, or both. A conversation cannot proceed without referencing a valid TCT entry.

All transaction identifiers and transaction program names (TPN) must be defined in a TCT to relate each transaction identifier to the appropriate TPN. Therefore, when a program issues an allocation request it uses a transaction identifier to indicate which transaction program it wishes to communicate with. The system relates this identifier to the TPN.

The TPN is carried in the attach request sent to the remote domain. At the remote domain attach processing will use the TPN to start the correct program to service the request. The TPN is therefore known to both LUs that are to participate in a conversation. Transaction identifiers are only known locally to each LU.

## Defining Transactions

The DEFTRANS command is used to define a TCT. This command allows you to define transactions between your domain and another LU.

### *Example*

To define a transaction with a transaction ID of DBQUERY and a TPN of DB01, enter the following command:

```
DEFTRANS TRANSID=DBQUERY TPN=DB01
```

When an NCL procedure wants to invoke a transaction with a transaction ID of DBQUERY, the TCT entry is located and the associated TPN is extracted. The TPN and any other information specified in the TCT entry is sent in an attach request to the remote system specified by the NCL procedure on the &APPC ALLOCATE verb.

If you want the transaction to start the NCL procedure DBQRY01 in the remote system, define the name of the procedure by using the PROC operand of the DEFTRANS command. This TCT entry must reside in the remote system.

## Defining a Qualified Transaction

You can use the same transaction ID to invoke a different TPN name during certain circumstances, such as, testing a new version of an NCL procedure.

Do this by defining a qualified transaction entry in the TCT and specify the same qualifier in the DLT entry that defines the remote LU to the local domain. The qualifier is specified by the QUAL operand of the DEFTRANS and DEFLINK commands. See the section, *Example 4—Running a Qualified Transaction*, on page 7-27, for a detailed example.

## Generic Transactions

A transaction identifier can have the same common suffix as the name of the transaction procedure it subsequently invokes—for example, the transaction identifier DBUPDT and the procedure APPCUPDT have the suffix UPDT in common. If there are many transactions starting with a fixed prefix such as DB that invoke procedures with a fixed prefix such as APPC but with the same variable suffix, a single generic transaction definition can be used to service them.

Generic transactions are defined by specifying masks for the transaction IDs, TPNs, and procedures in the TCT entry. See the section, *Example 5—Running a Generic Transaction*, on page 7-28, for a detailed example.

## Specifying a Default Destination

The destination of an APPC transaction request can be specified by using the &APPC ALLOCATE verb or in the TCT definition. If no destination is specified on the &APPC ALLOCATE verb, the destination in the TCT is used. If no destination is specified on the &APPC ALLOCATE verb or in the TCT, a default destination of ENV=CURRENT is assumed.

If a transaction request is always serviced by the same destination, you can specify a default destination in the TCT definition. To specify a default destination in a TCT definition, specify one of the following on the DEFTRANS command:

- A link name using the LINK operand
- An LU name using the LUNAME operand
- A Management Services domain ID, using the DOMAIN operand
- The NCL environment using the ENV operand (for local conversations)

### *Example*

To define an APPC link with a transaction ID of DBQUERY, a TPN of DB01, and a link name of NMB, enter the following command:

```
DEFTRANS TRANSID=DBQUERY TPN=DB01 LINK=NMB
```

For more information about the options that can be specified in a TCT, see the description of the DEFTRANS command in the *Management Services Command Reference*.

## Maintaining TCT Definitions

You can replace or delete a TCT definition by using Management Services commands. The REPTRANS command allows you to replace an existing TCT definition with new values. The DELTRANS command allows you to delete a TCT table definition. For more information, see the *Management Services Command Reference*.

## Defining a Dynamic Link Table (DLT)

A DLT is used to support the dynamic addition of different types of links, including INMC and APPC links. In the case of APPC the DLT is used to define those LUs with which the local system will allow automatic link activation. Two types of APPC links can be defined, those that support parallel LU6.2 sessions and those that support only a single LU6.2 session.

To specify a DLT, use the DEFLINK command.

*Example*

To allow an APPC link with an LU of NM1, enter the following command:

```
DEFLINK TYPE=APPC LUNAME=NM1 LINK=NM1
```

The DLT supports other options including the specification of a password for link level security. For more information, see the section, *Implementing Security for APPC Links*, on page 7-16.

### Specifying an Option Set

To include support for option sets, specify an OSCT definition with the OPSET operand. When an APPC link is activated, the OSCT is then accessed to determine the optional features it is to support.

If an option set is not nominated or the one nominated can not be found during link activation, the system uses a default option set. The default defines a parallel session link with no data mapping support and no conversation level security support for the connected system.

### Maintaining DLT Definitions

DLTs are maintained by using the REPLINK and DELLINK commands. REPLINK allows you to replace a DLT definition. DELLINK allows you to delete a DLT definition. For more information, see the *Management Services Command Reference*.

## Defining an Option Set Control Table (OSCT)

An OSCT is used to define the optional features that the APPC link will support during communication. These optional features are those provided by option sets.

To define an OSCT, use the DEFOPSET command. You can specify single or parallel session links by using the PARSESS operand. However, if not specified, the default is parallel sessions.

During the activation of a link, the OSCT entry nominated in the DLT definition for the destination is used to determine the options to be supported and to indicate mode names associated with the link.

*Example*

To define an OPSET of NMOPSET with single session links issue the command shown below:

```
DEFOPSET OPSET=NMOPSET PARSESS=NO
```

For more information, see the description of the DEFOPSET command in the *Management Services Command Reference*.

### Nominating Mode Names

An OSCT entry can be associated with up to four mode names. You can use the MODE operand to specify the names of the MCT entries that define the LU6.2 mode names associated with this OSCT entry. Multiple mode names are useful for parallel session links to group sessions with similar characteristics. If no mode names are specified in the OSCT definition, a default mode is used during link activation.

### Maintaining OSCT Definitions

OSCTs are maintained with the REPOPSET and DELOPSET commands. Use the REPOPSET command to replace an OSCT definition and the DELOPSET command to delete an OSCT definition. For more information, see the *Management Services Command Reference*.

## Defining a Mode Control Table (MCT)

To request a session with certain characteristics specify a mode name on the allocation request or an MCT definition in the OSCT. The MCT definition specified contains an LU6.2 mode name which has associated session characteristics.

To define an MCT entry, use the DEFMODE command.

*Example 1*

To define an MCT entry with the name LU62PARL which references the LU62PARL mode name, enter the following command:

```
DEFMODE MODE=LU62PARL MODENAME=LU62PARL
```

*Example 2*

To define an MCT entry without the MODENAME operand enter the following command.

```
DEFMODE MODE=LU62PARL
```

The name of the MCT entry is used as the LU6.2 mode name .

## VTAM Logmodes

To associate an LU6.2 mode name with specific session characteristics (for example, Class of Service, session level pacing), some implementations of APPC require the matching VTAM logmode names with the derived characteristics to be defined to the MCT via the LOGMODE parameter. If omitted, the VTAM default logmode values for those parameters are used.

**Note**

Management Services can successfully initiate sessions using LU6.2 mode names with or without matching VTAM logmode names. However, other implementations, such as CICS and OS/2, require the LU6.2 mode name to match the VTAM logmode name to successfully activate sessions initiated by these products.

## Specifying Modes for Single Session Links

If you are using single session links, the primary use of the MCT entry is to derive a VTAM logmode name if session establishment is initiated by Management Services.

## Specifying Modes for Parallel Session Links

If you are using parallel session links, mode names can be used to partition the available sessions into logical groups, with slightly different characteristics.

### *Example*

To define a parallel session mode that supports a maximum of ten sessions, a minimum of two contention winner sessions for the local domain, and a minimum of two contention winner sessions for the remote system enter the following command.

```
DEFMODE MODE=LU62PARL SESSLIM=10 MINWIN=2 MINLOSE=2
```

## Default Modes

If you do not nominate an option set in the DLT definition, both the default OSCT and the default MCT are used at link activation. The default MCT is the first entry in the MCT or if no entries are defined an internal default applies with MODENAME=APPCMODE, SESSLIM=20, MINWIN=5 and MINLOSE=5, no logmode name is defined. The same applies if you nominate an MCT entry in your OSCT but the nominated modes can not be found.

## Maintaining MCT Definitions

MCT entries are maintained with the REPMODE and DELMODE commands. The REPMODE command allows you to replace an MCT entry. The DELMODE command allows you to delete and MCT entry. See the *Management Services Command Reference* for more information about these commands.

## Implementing Security for APPC Links

There are two levels of security that can be provided for APPC links:

- Link level security, or LU-LU verification
- Conversation level security

The following sections describe these two security levels.

### Link Level Security

Link level security is a session level security protocol exchanged during session activation in order to confirm the identity of the partner LU. This requires the use of an eight character password known to both LUs that comprise the link. The password is not transferred on the session but used to encrypt data. The encrypted data is analyzed by the other LU to determine whether the password is correct or not.

The password is any hexadecimal string, and may be defined in the DLT, or supplied by an operator on a LINK START request. When a password is present Management Services invokes LU-LU session verification processing. If verification fails for any reason then link and session activation fails.

*Example*

To specify a password of A1B2C3D4 in a DLT definition for SYSB, enter the following command:

```
DEFLINK LINK=B LU=SYSB TYPE=APPC PASSWORD=A1B2C3D4
```

If PASSWORD is not specified when defining the DLT and a password is not given in the LINK START command, LU-LU verification is not performed.

### Conversation Level Security

Conversation level security is used to verify the identity of the partner transaction program. The relevant security information (for example, user ID and password) is sent in the attach request and is verified by the receiving LU. If the correct security information is not supplied the request is rejected.

A TCT is used to define the level of conversation security required by a given transaction. When an NCL procedure issues an allocation request the TCT entry is used to determine the level of security information that needs to be included in the attach request to be sent to the remote LU.

## Running a Secured Transaction

To run a secure transaction define the SECURITY operand of the DEFTRANS command in a TCT entry. Options on the operand are: NONE, SAME, USER, and USERPSWD. The default is NONE which means that the transaction runs without security.

*Example*

To secure the transaction for DBQRY01 with the user's password, enter the following command:

```
DEFTRANS TRANSID=DBQRY01 TPN=DB01 SECURITY=USERPSWD
```

When an allocation for the DBQRY01 transaction is issued, the password for the user environment is passed to the destination for verification.

### Security Options

If you specify SECURITY=SAME in the TCT, the allocation assumes the security of the request source. If you specified the user ID and password on the allocation request, it is assumes the same as if you had specified SECURITY=USER. If you do not specify any user information, it assumes the same as if you had specified SECURITY=NONE.

For allocation requests, security is performed for a transaction under the following conditions:

- The NCL procedure which issued the request is running in a signed-on environment
- The USERID and PASSWORD operands were specified in the allocation request

Otherwise it will run as an unsecured transaction.

Incoming attach requests that contain security information result in a secured transaction and if no security information is present the transaction runs as unsecured.

## Conversation Level Security and NCL Procedure Environments

NCL procedures running in a signed-on user environment can issue allocation requests for secured transactions without specifying their user ID and password. If an NCL procedure is running in an environment that is not protected by a user ID and password signon, such as EASINET, it can only request allocation of secured transactions by specifying the USERID and PASSWORD operands in the allocation request.

For incoming attach requests the TCT security level determines the type of NCL environment in which the NCL procedure nominated to service the transaction is run. An attach request for an unsecured transaction results in the NCL procedure being started in the background server (BSVR) region.

If the request is for a secured transaction a special APPC region is created (if none already exists) for the user within Management Services. The user is signed on using normal security procedures. This happens before the transaction can be started.

Secured transactions require a signed-on environment before execution is permitted. An attempt to start such a transaction in some other environment fails with a security error return code.

## Specifying Security for APPC Links with Remote Systems

The OSCT is used to indicate the security options that can be accepted from a remote LU. To specify security parameters, use the SECURITY operand of the DEFOPSET command. The options for the operand are: NONE, USER, and USERPSWD.

If you specify NONE, no security information sent by the remote LU in an attach request is processed by the local system; if present it is ignored. This implies that only unsecured transactions can be invoked by the remote LU.

If you specify USERPSWD, user IDs and passwords are accepted from the remote LU and passed to the local security system for verification. However, the already verified indicator is not accepted.

If you specify USER, then USERPSWD is assumed and in addition the already verified indicator is also supported. If the already verified indicator is set, a request for user ID authorization is still made to the local system to validate the region without a password.

### APPC Region Use

NCL procedures, defined in the TCT to service secured transactions, execute in a special APPC region built-in response to an attach request for a specific user ID. Normally all attached requests for the same user ID execute in the same APPC region. When the last NCL procedure completes execution the region is deleted.

To restrict the building of an APPC user region to a given user ID, specify a DLT entry using the USERENV operand of the DEFLINK command. The options for the operand are GLOBAL or LINK.

If GLOBAL (the default) is specified, an APPC region is built to service all attach requests from the same user ID from any remote LU.

If LINK is specified, an APPC region is built to service attach requests, for a given user ID, received from the remote LU defined by the DEFLINK definition.

## Activating APPC Links

You can activate a link in the following ways:

- Manually—via a LINK START command
- Automatically—as the result of an allocation request

### Starting Links Manually

To start an APPC link manually use the LINK START command.

*Example*

To start an APPC link from SYSA, issue the following command:

```
LINK START=NMB TYPE=APPC
```

### Deactivating Links Manually

Regardless of the way an APPC link is started it remains active while any sessions are active even when all conversation activity ceases. If all sessions on the link have closed then the link will automatically deactivate.

To stop the link at any other time, use the LINK STOP command. You can specify different options (DRAIN, QUIESCE, and FORCE) to influence the way the link deactivates.

If DRAIN is used (the default), all conversation activity is serviced before closing the link. If QUIESCE is used, active conversations are allowed to complete normally, but pending conversations complete with an allocation error. If FORCE is used, all conversations are terminated (with a resource failure return code) and link deactivation proceeds immediately.

*Example*

To deactivate the link to NMB by using the LINK STOP command with the DRAIN option, enter the following command:

```
LINK STOP=NMB TYPE=APPC
```

## Starting Links Automatically

A request to establish an APPC link may come from:

- The remote LU
- The local domain via a conversation request
- An operator request

Provided the correct definitions are in place any program issuing an allocation request can cause link activation if the link is not already active.

The LU name of a network node must be defined to Management Services in the DLT to authorize automatic LU6.2 session activation between the remote LU and the local domain. See the section, *Defining a Dynamic Link Table (DLT)*, on page 7-12 for information on how to define a DLT.

### Session Initiation Requests

For a link to be activated automatically, a request to initiate an LU6.2 session must come from a source in the network. All requests can be categorized as coming from one of three sources:

- An external logon request
- An external BIND request
- An internal NCL procedure request

Whenever the first session between Management Services and some other network node is requested, the Dynamic Link Table is examined to determine whether the link is authorized, and establish any special options for the link.

### *External Logon Request*

An external logon request arises when these three conditions occur:

- An LU in the network requests a session between Management Services and either itself, or some other LU
- Management Services is to be the primary session partner
- The logmode specified contains LU6.2 session parameters

Successful processing of such a request results in Management Services issuing a BIND to the target LU. The target LU can accept the BIND as it is, negotiate certain BIND parameters, or reject the BIND outright. If not rejected, the negotiated (or unchanged) BIND response is returned to Management Services and if acceptable the session establishment is complete.

### *External BIND Request*

An external BIND request arises when these three conditions occur:

- An LU in the network requests a session between Management Services and itself
- Management Services is to be the secondary session partner
- The BIND contains LU6.2 session parameters

Successful processing of such a request results in Management Services accepting (or negotiating) the BIND from the LU, and the session is established.

### *Internal NCL Procedure Request*

The internal NCL request, &APPC ALLOCATE, can be sourced from any executing NCL procedure to request an APPC conversation with some remote LU. If no suitable sessions are available to that LU, and the session limit (as defined in the MCT) has not yet been reached, then Management Services issues a session setup request to VTAM using the logmode name from the MCT, whereupon session initiation proceeds as for an external logon request above.

**Note**

Wherever Management Services initiates a session it is the primary session partner, and no logmode is required as Management Services formats the LU6.2 BIND according to the options for the link.

Sessions where Management Services is the secondary session partner must be initiated by the other (primary) session partner. This does not restrict any session usage as it is a peer relationship, and aspects regarding session polarity (such as which LU is contention winner) are negotiable.

Regardless of the source of the session initiation request, and whether Management Services is to be the primary or secondary session partner, session establishment proceeds through a number of phases as described below.

**Note**

If a session initiation request comes from an &APPC ALLOCATE verb, Steps 4 and 5 are performed first, before the session setup request is passed to VTAM. Following this, and for all other session initiation types, processing begins at Step 1 when VTAM notifies Management Services of the arrival of a logon request or BIND request.

1. Session parameter validation ensures that the BIND to be sent (when processing a logon request), or BIND received (when processing an unsolicited BIND), is a valid LU6.2 BIND within the parameters supported by Management Services.

   If necessary, the BIND parameters are modified before being sent to the secondary LU, or negotiated with the primary LU, so that they are within the bounds supported by Management Services.

2. The unqualified network LU name provided by VTAM is extracted. If other LU6.2 sessions already exist with this network node, or are in the process of being established, then Step 3 is performed; otherwise Step 4 is performed next.

3. Where other LU6.2 sessions exist, or are activating, to the destination involved, additional checks are made to determine whether the new session is acceptable within the session limits determined by the MCT entries (see Step 5 below). If so, the session is accepted as a parallel session without further processing; otherwise it is rejected.

4. Where no other LU6.2 sessions exist, and none are being activated, to the destination involved, the unqualified network LU name is used to locate a matching entry in the DLT. If an entry cannot be located the session initiation is rejected. Otherwise the DLT details are extracted and used to initialize the APPC link connection. If an OSCT entry was nominated in the DLT, it is located and the processing options are extracted.

5. The OSCT entry determines whether this link supports parallel sessions or not. Any mode names listed for the OSCT entry are used to locate the corresponding MCT entries, and the details of each MCT is extracted to form a valid MODE name list for this destination. For each mode name, this list contains the session limits for that mode, as well as other session control information.

### *Single Session Links*

When the OSCT entry indicates that parallel sessions are not supported, an external BIND request to establish a parallel session link is rejected. Only a single session link can be established if the OSCT entry has been defined with PARSESS=NO. Hence only one conversation at a time can be active on such a link, and other conversation requests must queue, pending session availability.

If the session setup request comes from the local domain (via a LINK command or NCL procedure allocate request), Management Services attempts to set up a contention winner session. However, the polarity of the session can be negotiated by the secondary LU. If the session setup request comes from outside the local domain, Management Services accepts the session polarity without negotiation.

### *Parallel Session Links*

When the OSCT entry indicates that parallel sessions are supported, an external BIND request to establish a single session link is accepted, and a single session link is set up. An external BIND request to establish a parallel session link is accepted and session limits negotiation is initialized.

For a parallel session link, Management Services initializes the link session limit as the sum of the session limits for each mode defined for the link. An attempt is made to initially activate sessions within each mode to at least reach the minimum contention winner counts for the mode. Additional sessions may be activated if conversation demand cannot be immediately serviced by the available sessions.

During link initialization a special transaction known as Change Number of Sessions (CNOS) is executed for parallel links. This transaction negotiates the session limits for the defined modes before sessions are established. The CNOS transaction may be subsequently invoked on demand, such as by remote LU request, to renegotiate session limits at any stage of link operation.

## Session Selection for Conversations

When an allocation request targets a particular link an attempt is made to locate a session to be used by the conversation. If an idle contention winner session is located then it is assigned. If not, the conversation goes into a pending allocation state, awaiting a session for the conversation.

For a pending allocation on a parallel session link, further efforts are made to isolate a usable session. Where the number of contention winners can be increased (according to the session limits) then a new session is started. Otherwise, if an idle contention loser session is located, a bid is made to use that session.

For a pending allocation on a single session link, if the session is a contention loser session and it is idle, a bid is made to use that session.

In general, when a conversation enters a pending allocation state, one of three events can occur to provide a session for its use:

- An active conversation ends, freeing up a contention winner session for the pending conversation.
- A new contention winner session starts and becomes available for the pending conversation.
- A response to previous bid indicates that the remote end will allow the conversation to commence on a contention loser session.

Conversations remain pending until one of these events frees a session for use, or until the allocation request is otherwise cancelled. An allocation may time-out (if a timeout period is specified on the allocation), or a link termination condition may cause the cancellation of all pending requests for the link.

### Deactivating Links Automatically

APPC links deactivate automatically when all sessions on the link have been closed.

## APPC Link Definition Examples

The following examples illustrate typical APPC table definition requirements to establish LU6.2 connections between the following:

- Two domains (SYSA and SYSB)
- Two domains (SYSA and SYSB) that require security verification
- Management Services and a remote LU
- Two domains (SYSA and SYSB) using a qualified transaction
- Two domains (SYSA and SYSB) using a generic transaction

Each example uses a database query application with transaction identifier DBQUERY, a TPN of DB01, and a server NCL procedure DBQRY01.

When defining mode names in the examples, the MODENAME parameter has been omitted in the DEFMODE command, so that the LU6.2 mode name defined is the same as the name of the MCT entry (MODE parameter).

## Example 1—Running Conversations Within the Same Domain

The two LUs of an APPC conversation are run in the same domain to develop a database query application and test it. No sessions are established, no session related definitions are needed, and session over heads are eliminated. Only the TCT entry needs to be defined, with the name of the local domain as the destination LU.

To define an APPC link within the same domain, SYSA, perform the following steps:

Step 1. Define the following TCT entry in SYSA:

```
DEFTRANS TRANSID=DBQUERY TPN=DB01 LU=SYSA PROC=DBQRY01
```

Step 2. Issue an allocation request from an NCL procedure in SYSA for DBQUERY.

The allocation processing locates the TCT entry and determines that the destination is the local domain. This has the effect of starting DBQRY01 in SYSA. During this processing there is no need to reference any of the session related tables DLT, OSCT or MCT.

## Example 2—Running a Link Between Two Domains with Already Verified Security

An APPC link is to be established between two domains SYSA and SYSB. The link is to service transactions that may require data mapping support.

To define a link between SYSA and SYSB with data mapping support, perform the following steps:

Step 1. Define the following DLT entries in SYSA and SYSB that specifies an OSCT entry that supports data mapping:

```
DEFLINK LINK=L01 LU=SYSB TYPE=APPC OPSET=NMMAP (in SYSA)
DEFLINK LINK=L02 LU=SYSA TYPE=APPC OPSET=NMMAP (in SYSB)
```

Step 2. Define the following OSCT entry, NMMAP, in both domains:

```
DEFOPSET OPSET=NMMAP MODE=MODELU62 SEC=USER MAP=YES +
LOG=YES PARSESS=YES
```

SEC=USER indicates that the already verified indicator is to be accepted from the remote domain

**Note**

If you want the link to be a single session link, specify the PARSESS=NO operand and a MODE that supports single session links in the OSCT entry.

Step 3. Define the following MCT entry, MODELU62 in both domains:

```
DEFMODE MODE=MODELU62 SESSLIM=4 MINWIN=1 MINLOSE=0
```

The LU6.2 mode name defined is MODELU62 which supports a maximum of 4 sessions.

Step 4. Define the following TCT definitions in each domain to allow the invocation of the DBQURY01 NCL procedure:

```
DEFTRANS TRANSID=DBQUERY TPN=DB01 LINK=L01
MODENAME=MODELU62 (in SYSA)

DEFTRANS TRANSID=DBQUERY TPN=DB01 PROC=DBQRY01 (in SYSB)
```

A default mode name for conversation allocation has been specified.

Step 5. An NCL procedure in SYSA issues an allocation request for DBQUERY.

The TCT entry is located and an attach request for DB01 built. The link name L01 in the TCT is used to locate the DLT and to determine the LU name of the destination, SYSB. The OSCT entry NMMAP nominated in the DLT and the corresponding MCT entry MODELU62 nominated in the OSCT are also located. If the link is not already active the DLT entry is used to start link activation and a parallel session link with data mapping support is established.

Once the link is active and a session is available the attach request for DB01 is sent to SYSB. Attach processing finds the TCT entry for DB01 and the NCL procedure, DBQRY01 is started.

## Example 3—Running an APPC Link Between a Domain and a Remote LU

An APPC link is to be established between a domain and an OS/2 work station. The link is to service transactions with no data mapping requirements.

OS/2 requires that the mode name match a valid LU6.2 VTAM logmode, therefore in addition to a DLT, you need to define an OSCT and its associated MCT.

To run an APPC link between your domain and an OS/2 workstation, complete the following steps:

Step 1. Define the following DLT in your domain:

```
DEFLINK LINK=WRKS01 LU=OS2001 TYPE=APPC OPSET=WRKSOPST
```

WRKSOPST is the name of the OSCT entry to be used.

**Note**

Ensure that you enable APPC links in your OS/2 system.

Step 2. Define the following OSCT entry in your domain:

```
DEFOPSET OPSET=WRKSOPST MODE=LU62SESS LOG=YES MAP=NO
```

LU62SESS is the name of the MCT entry to be used.

**Note**

If you want the link to be a single session link, specify the PARSESS=NO operand and a MODE that supports single session links in the OSCT entry.

Step 3. Define the following MCT entry in your domain:

```
DEFMODE MODE=LU62SESS LOGMODE=LU62SESS SESSLIM=20 +
MINWIN=1 MINLOSE=0
```

The LU6.2 mode name defined is LU62SESS and supports a maximum of 20 sessions. A logmode has been specified and is the same as the LU6.2 mode name LU62SESS. OS/2 requires mode names to match a valid VTAM logmode entry.

Step 4. Define the following TCT entries so that the DBQUERY transaction can be invoked from the workstation and the database interrogated in SYSA:

```
DEFTRANS TRANSID=DBQUERY TPN=DB01 PROC=DBQRY01
```

Step 5. A program running in the workstation issues an allocation request for TPN DB01.

Your domain receives an LU6.2 session request from LU OS2001. The LU name is used to locate the DLT for LU OS2001. The associated OSCT and MCT are also located. A parallel session link with no data mapping support is established. The workstation sends an attach request for DB01 to your domain. The TCT for DB01 is located by your domain and the procedure DBQRY01 is started to service the transaction.

## Example 4—Running a Qualified Transaction

A database query application runs between two domains SYSA and SYSB. The DBQUERY transaction is invoked from SYSA to send an attach request for DB01 to SYSB, where the NCL procedure DBQRY01 is run. A new version of this procedure (DBQRY01T) needs to be tested using TPN DB01TEST and the transaction qualifier TEST.

Perform the following steps to use the APPC link to test the new NCL procedure:

Step 1. Define the qualifier TEST in the DLT entry in SYSA by issuing the following command:

```
DEFLINK LINK=B LU=SYSB TYPE=APPC QUAL=TEST
```

Ensure that you stop and restart the link to put this qualifier into effect.

Step 2. Define the qualified entry in the TCT entry in SYSA by issuing the following command:

```
DEFTRANS TRANSID=DBQUERY TPN=DB01TEST QUAL=TEST
```

Step 3. Define a TCT entry in the remote system (SYSB) for the procedure being tested by issuing the following command:

```
DEFTRANS TRANSID=DBQUERY TPN=DB01TEST PROC=DBQRY01T
```

Step 4. Issue an allocation request for DBQUERY.

The existing unqualified TCT entry is used to resolve the name of the destination. The destination name is used to determine that the qualifier TEST is in effect (as defined in the DLT). The TCT entry for DBQUERY qualified by TEST is located and used to complete the allocation request. An attach request for DB01TEST is sent to SYSB where the TCT entry for DB01TEST is found and DBQRY01T is run.

## Example 5—Running a Generic Transaction

Two domains SYSA and SYSB are connected via an APPC link. Transactions starting with the prefix DB (for example, DBQUERY, DBPUT, DBUPDT) are to be allocated in SYSA. The NCL procedures to service these transactions start with the prefix APPC (for example, APPCQRY, APPCPUT, APPCUPDT). This can be done using a single generic TCT entry.

To define a single TCT entry to service all the above transactions, perform the following steps:

Step 1. Define the following TCT entry in SYSA:

```
DEFTRANS TRANSID=DB* TPN=TP* LINK=B
```

Step 2. Define the following TCT entry in SYSB:

```
DEFTRANS TRANSID=DB* TPN=TP* PROC=APPC*
```

Step 3. Issue an allocation request in SYSA for transaction ID DBUPDT.

This matches the TCT entry DB*. The matching characters in the transaction name are removed and the remaining characters (UPDT) are appended to the generic TPN (TP*) to get TPUPDT. This is the TPN sent to the remote domain. It matches the TCT entry for TP* and again the matching characters are removed. The remaining characters UPDT are appended to the generic procedure name APPC* to form the NCL procedure name APPCUPDT to be started.

## APPC Application Programming Facilities

APPC application programming facilities are described in the *Network Control Language User's Guide*.

# 8

# Partitioning VTAM Networks

This chapter describes how to use the Network Partitioning Facility (NPF) to partition your VTAM networks.

**This chapter contains the following topics:**

- About the Network Partitioning Facility (NPF)
- Storing Network Partitioning Information
- VTAM Message Table Requirements
- Defining NPF Control Members
- Defining NPF Resource Tables
- Defining Network Partitioning for the System Console
- Using NPF when Logging on to Remote Systems

## About the Network Partitioning Facility (NPF)

NPF allows you to allocate control of sections of your network to different network operators. By subdividing your network into partitions, operators influence and receive details only about the partition for which they are responsible. NPF is useful in the following circumstances:

- In a large network where you want operators to control different sections of the network
- In smaller networks where you want some operators to control a test network, while other operators are responsible for a production network
- In a mixed development and production environment where you need programmers to control a group of terminals for testing purposes

NPF allows you to restrict operator access to the resources in your network in two ways:

- Command partitioning—specifies the resources that operators can control by issuing commands
- Message partitioning—specifies resources from which operators receive unsolicited messages

These restrictions are described further in the following two sections, *About Command Partitioning* and *About Message Partitioning*.

NPF can also be extended for use by NCL applications to partition or restrict access to non-network resources.

### About Command Partitioning

Command partitioning determines the VTAM resources which an operator can control by issuing VTAM commands. These resources are defined in one or more resource definition tables. The tables can include resources that overlap or copy those defined for message partitioning.

If an operator attempts to execute a VTAM command referencing a resource that is not within their defined command partition table, the command is rejected and a message displayed. If no command partitioning is defined for a user, then any VTAM resources can be referenced.

**Note**

Where common tables are shared by both command and message partitioning, any initial status applies only to message partitioning and is ignored for command partitioning.

When command partitioning is in place for an operator, only VTAM commands that reference a resource name are affected. Operators can issue specialized commands, such as D NET,BFRUSE, on any resource within the network.

The use of the VTAM display (D) command is also restricted to the resources defined within the user's command partitions. However, the SYSPARMS VDISPLAY operand can be used to allow a user to display any resource within the network. See Appendix D, *SYSPARMS Operands*, for more information about VDISPLAY.

## About Message Partitioning

Message partitioning defines the resources within the network for which an operator can receive unsolicited (PPO) VTAM messages. The resources for which an operator is to receive unsolicited messages are defined in one or more resource definition tables. These resources can overlap or copy those covered by the same user's command partitions.

### Resource Table Status

Each resource table is given an initial status of either active (A) or inactive (I). The status of a resource table can vary from user to user. The status of a resource table can be changed online by using the NPTAB command. See the *Management Services Command Reference* for details of this command.

When a table is active for a user, the resources within that table are used to determine whether the user is to receive a particular unsolicited message. For example, if an unsolicited message arrives that refers to a resource defined in that table, the message is sent to the user's terminal.

**Note**

If you specify a resource table that includes both command and message partitioning, any initial status applies only to message partitioning and is ignored for command partitioning.

### Assigning to Multiple Operators

It is recommended that you assign the same message partitioning to many operators. This ensures that control can be transferred to meet the requirements of geographical boundaries, time zones, network reconfiguration, or other commitments. Multiple operators can also be defined to receive messages for the same resources.

Message partitioning works in conjunction with other options that control the delivery of unsolicited messages. It does not affect the receipt of undeliverable messages if the user is profiled to receive them.

These options are detailed in the section titled *Defining a User ID*, on page 3-6.

## Storing Network Partitioning Information

Network partitioning information is stored on the NPF dataset. This dataset differs depending on the operating system that you have. Control members and resource tables have the same format for all operating systems.

### Storing Network Partitioning Information in OS/VS Systems

In OS/VS systems the NPF dataset is a partitioned dataset identified by the NPTABLES DD statement in the JCL for the system. The members in this PDS should contain fixed length 80-byte records.

This dataset contains both control members and resource tables. It can be updated online without any need to restart the system.

### Storing Network Partitioning Information in VM Systems

In VM systems the NPF dataset is a CMS minidisk identified by the NPFFM startup parameter. This parameter designates the CMS FILEMODE associated with NPF command and resource table definitions. The NPFFT startup parameter also defines the CMS FILETYPE associated with NPF members.

Individual NPF members are regarded as individual CMS files on the minidisk.

For example, an NPF command member CM001 resides on the NPF minidisk as:

```
CM001 NPF *
```

where NPF and * are the default values for these parameters. The default file mode * signifies that all Management Services minidisks are searched for the file CM001 NPF.

## VTAM Message Table Requirements

There are requirements for VTAM message tables when using NPF. The requirements vary depending on your operating system.

### VTAM Message Tables in OS/VS Systems

The resource names in VTAM messages need to be at fixed offsets in the messages. For this to occur, the USSMSG macros in your installation's ISTINCNO must specify OPT=NOBLKSUP. If this is not the case, then you should alter the message table, and reassemble and relink it.

### VTAM Message Tables in VM Systems

In VM systems, create a copy of the VTAM message table, by using the link editor, and place it in a load library accessible to Management Services. The load module must be named NMISTCMM.

Link edit statements for the VMFLKED command to perform this link are provided in the distributed file NPFTABLE LKEDCTRL. Input to the link edit is the VTAM object module ISTCFCMM TXT. The %LIBRARY control card must point to a load library defined within the GCS startup procedures for Management Services by a GCS GLOBAL LOADLIB command.

Full details of the link edit procedure are provided in your installation instructions for VM systems.

NPF uses this table during system initialization only. Native VTAM message tables are not referenced or modified.

This link edit must be repeated whenever a new version of VTAM is installed on your VM system.

The USSMSG macros in your installation's ISTINCNO must specify OPT=NOBLKSUP.

## Defining NPF Control Members

To specify command and/or message partitioning for a user, specify a control member in the NPF Command Member and/or NPF Message Member fields of their user ID definition.

The control member is the name of a member of the NPF library. The member defines a list of other members that are the resource definition tables specifying the resources the user can control and receive messages from.

### Combined Control Members

You can specify an installation defined control member that holds both the command and message partitioning for the user ID. It can also contain other installation defined restrictions. This member is interrogated by applications using the &RSCCHECK built-in function. This member is specified in the NPF Resource List Member field of a user ID definition.

If you specify an NPF Resource List Member, you cannot specify a member in the command and message partitioning fields. Also, if you specify NPF message and command members, you cannot specify an NPF resource list member.

Figure 8-1 shows the relationship between the UAMS user ID definition, the resource list control member, and resource tables.

### Coding Control Members

Control member statements have the form:

```
resource-group=qualifier.resource-member[,{ I | A }]
```

where:

**resource-group**

The name of the class of resources to which elements in the resource-member belong. The name can be any valid 1 to 8 character name. The following resource groups are specified by Management Services for storing message and command partitioning:

$NMMSG Message partitioning

$NMCMD Command partitioning

Names prefixed with a dollar sign ($) are reserved for Management Services use.

***qualifier***

An optional 1- to 8-byte resource qualifier which applies to all resources specified in the resource member. For the $NMMSG and $NMCMD groups the qualifier represents the network name for the network resources (if not specified it defaults to the network name for the Management Services domain). You can specify an asterisk (*) to indicate unrestricted qualified access to the resource table.

***resource-member***

The NPF dataset member name which contains the resource names and masks for the resource group to which the user is to have access.

**,I**

Specifies that this member is inactive for the user. This option is not valid for the $NMCMD resource group.

**,A**

Specifies that this member is active.

## Syntax Rules for Control Member Statements

Only one control statement is allowed per line. Anything following the first blank in the control statement is treated as a comment.

**Note**

A control statement must commence in column 1.

The following are all treated as comments:

- Control member lines with an asterisk (*) in column 1
- All data after a semicolon (;)
- Blank lines

Resource-members can be used for multiple resource groups.

It is recommended that you use a naming convention that allows control members to be easily identified.

**Note**

You can grant unrestricted access to a resource group to a user who is NPF-restricted, by specifying the following:

```
resource-group=*.*
```

*Examples*

To specify a control member for message partitioning in the network NET001 using the NPFNW01 resource table definition, code the following:

```
$NMMSG=NET001.NPFNW01,A
```

To specify a control member for command partitioning in the network NET001 using the NPFNW02 resource table definition, code the following:

```
$NMCMD=QNETIDA.NPFNW02
```

To specify a control member for the installation defined resource list, LIST01, in the network NET001 using the NPFNW03 resource table definition, code the following:

```
LIST01=NET001.NPFNW03
```

### Syntax Errors

When a user logs on, their defined control member is processed. If there is any syntax error in the control member, a message is written to the activity log identifying the statement in error, and the control statement is ignored. Processing continues with the next statement. When the user enters OCS, a warning message is displayed indicating that there is an error.

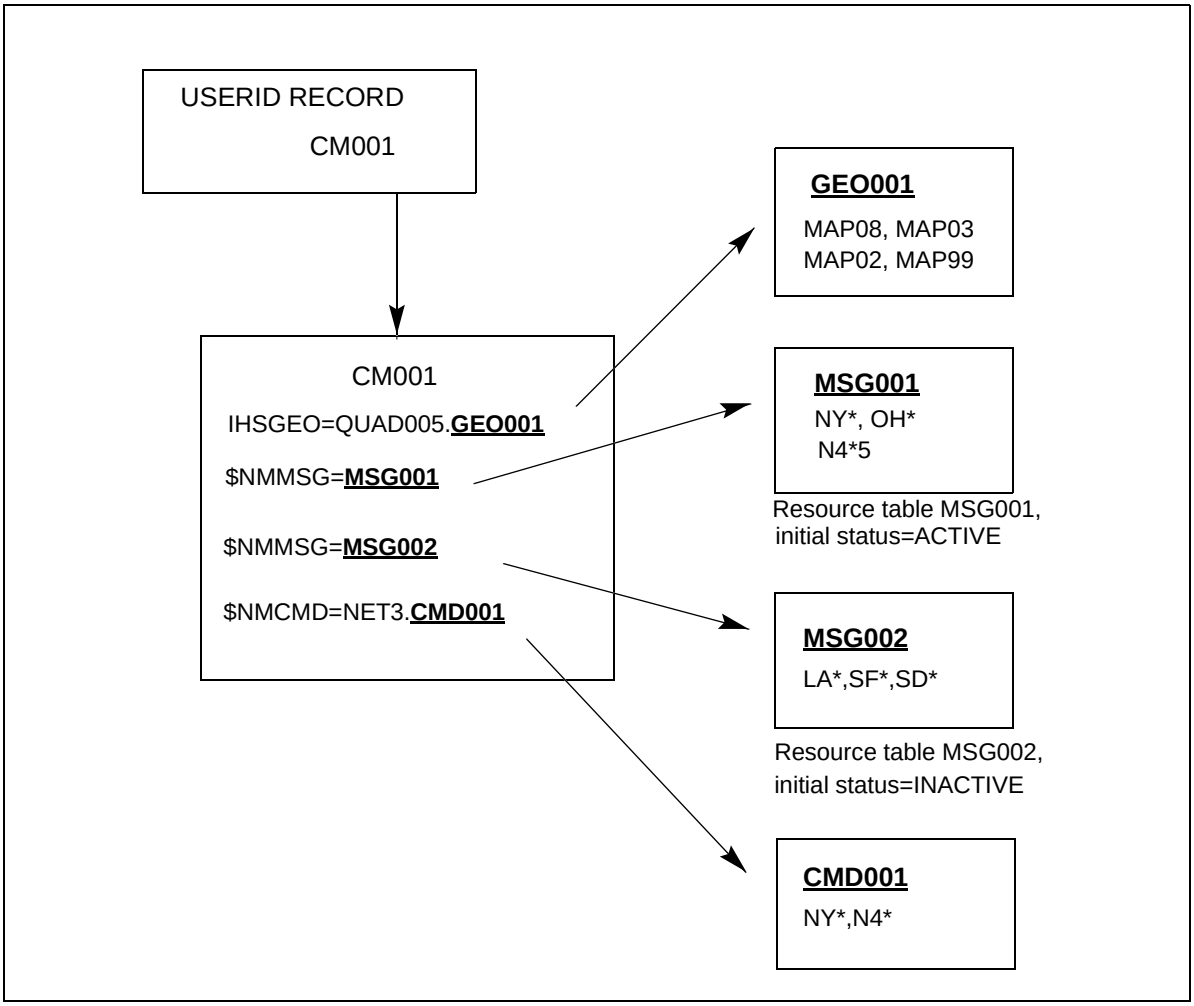
## Defining NPF Resource Tables

Resource tables are the members of the NPF dataset that define the groups of network resources for command and message partitioning. An unlimited number of resources can be defined. See Figure 8-1 for examples of resource tables.

The definition facility supports the use of generic definitions, using wildcard characters.

Resource tables can be updated online without having to restart the system. New tables can also be tested to ensure that they contain no syntax errors before committing them to live operation.

*Figure 8-1. Schematic of NPF Definition Requirements*




In the above figure, the dataset names CM001, GEO001, MSG001, MSG002, and CMD001 are all members of the NPTABLES dataset.

## Coding Resource Tables

Resource tables are coded as members of the NPF library and are referenced by the control member.

## Syntax Rules for Coding Resource Tables

The following syntax rules apply to the coding of resource tables:

- An asterisk (*) in column 1 indicates the entire line is a comment.
- A semicolon (;) anywhere on a line causes the rest of the line to be treated as a comment.
- Only columns 1 to 72 are processed. Sequence numbers in columns 73 to 80 are accepted and advised so that they can be used in error messages to help identify records in error.
- A record with blanks in columns 1 to 72 is ignored.
- The first non-blank in a record is treated as the start of a resource name and validation starts from that point.
- Multiple resource names can be specified in a record. They must be delimited by commas (,) as the first blank following a resource name terminates processing for that record, allowing comments to follow a resource name.
- Resource names cannot contain percent signs (%), single quotes (’), or double quotes (").
- If the first non-comment line in a member is %FORMAT=BASIC, or if there is no %FORMAT line, then the format used is the one applying to Management Services versions prior to Version 3.7, that is:
  - Names are restricted to a maximum of eight characters.
  - Names cannot contain question marks (?).
  - An asterisk (*) anywhere in a name, except at the end, matches any one character at that position.
  - An asterisk (*) at the end of a name means a generic match.
- If the first non-comment line in a member is %FORMAT=ADVANCED, then an advanced format is used:
  - Names can be from 1 to 64 characters.
  - A question mark (?) anywhere in a name matches any one character at that position.
  - An asterisk (*) anywhere in a name matches zero or more characters at that position.
  - The %ACCEPT and %REJECT statements, on a line by themselves, cause the following matches to be accepted (default) or rejected (which is a useful way of forcing off access to certain resources). The action at the end of a table is always REJECT.

For example:

```
%FORMAT=ADVANCED
L585,L6*,L7?5*
%REJECT
LAX*,NYE*
%ACCEPT
LA12*
```

It is recommended that you use a naming convention which lets you easily identify those members which are resource table members. Figure 8-1 includes examples for defining resources using wildcard characters.

## Loading Tables

When a user logs on with a user ID that is defined with NPF control members, the control member is loaded, scanned, and validated. A list of the tables required for that user is then built and matched against the resource tables that are already in the system. The following scenarios are possible:

- If a resource table is already available, then it is not reloaded, but made available to the requesting user.
- If a resource table is not already in the system, then it is loaded, validated, and made available to the requesting user.
- Any errors detected in this process are logged. When the operator re-enters OCS, messages are issued indicating any errors.
- If a resource table that does not exist is requested, the table entry is still built and classified as being available, but it is empty.

  To rectify this situation, you can create the missing member in the NPF dataset and then use the NPTAB REP command to replace the table entry with the newly created table. Users requiring this table have access to it immediately and do not need to log off and log on.

## Testing New Tables

To test newly-created tables before they are used by an operator, use the NPTAB ADD command. This command adds a new table to the system, validates it, and issues an error message if any errors are detected. All errors result in a message being written to the activity log. These messages name the member in error, and display the record sequence number, if they have been used.

Use the SHOW NP command to display any errors that occurred during control member or resource table processing. To provide an expansion of the table, use the SHOW NPTAB=*tablename* command. This identifies the records in error.

To delete a table once it has been tested, use the NPTAB DEL command.

## Dealing with Table Errors

If an operator encounters an error message when they log on, due to an error in their NPF environment, then they can use the SHOW NP command to list the resource tables available to the system. Any tables in error are flagged by an error indicator identifying the type of error that was detected. This error should be reported to you for investigation.

## Replacing Tables Online

Resource tables can be replaced online using the NPTAB REP command. Changes to the control member are not effective until you have logged off and logged back on.

## Deleting Tables

To delete a table that has been added by an NPTAB ADD command or by a control member reference, use the NPTAB DEL command. This command deletes the table immediately and any users referencing it can no longer use it.

If a table is deleted in error, it can be recovered using the NPTAB ADD command. However, users referencing the table when it was deleted must log off and log back on to regain use of the table.

If a table is deleted, then ensure that it is not referenced by any control members; otherwise the table is added again the next time an operator using the control member logs on.

## Displaying Tables

An operator can display the resource tables in their environment by using the SHOW NP command.

To display the list of the tables currently available to the system, use the SHOW NPTAB command. Any tables in error are highlighted.

To display the syntax errors in a specific resource table, use the SHOW NPTAB=*tablename*. All the resource names defined in the table are displayed and any errors are identified by a highlighted message at the point where the error occurs.

**Note**

The format of a table (BASIC or ADVANCED) is displayed. All resources listed are always displayed as if they were entered in ADVANCED format (names in BASIC format have embedded asterisks (*) changed to question marks (?)).

## Changing the Status of a Message Table

Message resource tables are assigned an active or inactive status when they are defined. To change the status of a message resource table, use the NPTAB command.

For example:

To activate a resource table that is currently inactive, enter the following command:

```
NPTAB ACT=ZONE4
```

To inactivate a resource table that is currently active, enter the following command:

```
NPTAB INACT=ZONE5
```

For details about any of the commands mentioned in this section, see the *Management Services Command Reference*.

## Defining Network Partitioning for the System Console

Message partitioning can be specified for the system console to help control the flow of unsolicited messages. Message partitioning is specified in the user ID associated with the system console (normally *pppp*OPER). Specifying command partitioning for the system console is of no value.

Other system users can also have partitioning. For example, the LOGPROC user can be partitioned to selectively log VTAM messages in the system log.

Any changes made to the NPF members for the *pppp*OPER user ID can be activated immediately by issuing a SIGNON command from the system console (for example, in MVS as F NETM,SIGNON). Changes made using the NPTAB REP command are also effective immediately.

## Using NPF when Logging on to Remote Systems

When a user logs on to a remote domain using ROF, the command partitioning and message partitioning that have been specified for that user on the remote domain are applied.

A user can have different command and message partitioning in different domains only if the systems use different UAMS datasets. If the UAMS dataset is shared between all domains, then a user has the same command and message partitioning no matter which domain they are logged on to.

# 9

# Customizing MAI-OC

This chapter provides information on how to customize MAI-OC.

**This chapter covers the following topics:**

- About MAI-OC
- Using System Parameters to Customize MAI-OC
- Using MAI-OC with Multiple Domains
- Using an MAI Installation Exit
- Defining MAI-OC to Target Applications
- Sample LU Definitions

For information on using MAI-OC, see the *Management Services User's Guide*.

## About MAI-OC

MAI-OC allows you to 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.

### Using MAI-OC

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.

An MAI-OC session can be operated from an NCL procedure using standard internal command environment processing through the &INTCMD facility. The NCL procedure can send data across the MAI-OC session(s) 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.

Before allowing users to use MAI-OC, you need to take into account the use of MAI-OC sessions with certain subsystems, and the use of MAI-OC from multiple domains to the same target applications.

Reference is made in this section to the use of some of the commands that control the operation of MAI-OC. The use and syntax of MAI commands are described in the *Management Services Command Reference.*

## Using System Parameters to Customize MAI-OC

This section describes the SYSPARMS operands which are used to customize MAI-OC. The SYSPARMS command is normally included in the INIT procedure, but can be issued at any time by an authorized user.

The following SYSPARMS command operands are applicable to MAI-OC.

```
SYSPARMS    [ MAIEX02=exitname | NO ]
            [ MAIEX02S=SYSTEM | USER ]
            [ MAIONL=YES | NO ]
            [ MAIOPREF=prefix | NMMAV ]
            [ MAIOTRNS=(xx,yy) ]
```

**MAIEX02=*exitname* | NO**

Specifies the name of an exit to provide security checking and restriction for MAI-OC sessions.

**MAIEX02S=SYSTEM | USER**

Specifies whether calls to the exit are synchronized at a system level, or synchronized at the user level.

**MAIONL=YES | NO**

Specifies whether MAI will append a new line character to data sent to a target application.

**MAIOPREF=*prefix* | NMMAV**

Specifies a prefix used to generate LU names for MAI-OC sessions. The default is NMNAV.

**MAIOTRNS=(*xx,yy*)**

Specifies values to alter the translate table used by MAI.

These operands are described in detail in Appendix D, *SYSPARMS Operands*.

## Using MAI-OC with Multiple Domains

When more than one domain 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 domain. This associates the names used by each domain with the domain in which that domain is executing, and avoids VTAM definition conflicts when attempting to start cross-domain sessions.

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

## Using an MAI Installation Exit

An MAI Installation Exit (MAIEX02) is provided with Management Services. This provides security checking and validates and/or 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 may override the LU name or prefix if required. Use of the exit is described in Appendix L, *MAI Installation Exit MAIEX02*.

## Defining 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 that MAI-OC will 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, and so on.

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. Remember that all LU names likely to be used to create a session with these applications should be defined as a separate terminal to that application. See the appropriate subsystem manuals for details on definition requirements and to tailor the definitions to your own requirements.

### Defining MAI-OC to a CICS/VS System

The following definition allows an MAI-OC session to be started with the 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
```

## Definition MAI-OC to an IMS/VS System

The following definition allows an MAI-OC session to be started with IMS 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=256

NAME        lterm
```

## Defining MAI-OC to a JES2 System

The following definition allows you to 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
```

## Defining MAI-OC to a JES3 System

The following definition allows you to 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
```

# 10

# Customizing Broadcast Capabilities

This chapter describes how to set up your requirements for sending broadcasts to multiple remote domains. It covers the following topics:

**This chapter contains the following topics:**

- About Customizing Broadcast Capabilities
- Listing System Group Definitions

For more information about broadcast services and how to use it, see the *Management Services User's Guide*.

## About Customizing Broadcast Capabilities

Broadcast services can be customized to send messages to one or more remote domains. This customization is performed by setting up system group definitions.

Each system group definition contains one or more remote domains to which a broadcast is sent. By specifying a system group definition when sending a broadcast, that broadcast is sent to the remote domains defined in that group definition. The local domain can also be included in the system group definition.

The following sections in this chapter describe how to create these system group definitions.

## Listing System Group Definitions

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, enter **/BCAST.LS** at a ===> prompt.

The Broadcast Services : Group List panel is displayed.

*Figure 10-1. Broadcast Services : Group List*

```
PROD------------------ Broadcast Services : Group List -----------------------
 Command ===>                                                Scroll ===> PAGE

                              S=Browse C=Copy D=Delete U=Update RS=Resources
                   Include
     Name          Local?  Description
     SYSGRP1       NO      System Group 1
     SYSGRP2       YES     System Group 2
     SYSGRP3       NO      System Group 3
     SYSGRP4       NO      System Group 4
     **END**









   F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
   F7=Backward  F8=Forward   F9=Swap                   F11=Right
```

If no system groups have been defined, no entries appear in this list. See the following section, *Adding a System Group Definition*, for details on how to add a system group definition.

For details of the information displayed and the actions available on the Broadcast Services : Group List panel, press F1 (Help).

## Adding a System Group Definition

To add a new system group definition, complete the following steps:

Step 1. Enter **/BCAST.LS** at a ===> prompt.

The Broadcast Services : Group List panel is displayed.

Step 2. Press F4 (Add).

The Broadcast Services : Group Definition panel is displayed.

*Figure 10-2. Broadcast Services : Group Definition*

```
PROD---------------- Broadcast Services : Group Definition ----------------
 Command ===>                                                   Function=ADD


 Group Name .............
 Description .............
 Include Local System? ...                 (Yes or No)









  F1=Help      F2=Split     F3=File      F4=Save     F5=Resources
                           F9=Swap                                F12=Cancel
```

Step 3. Type the name of the new system group in the Group Name field.

Step 4. Type a description of the system group in the Description field.

Step 5. To include the local system in this system group, type **Yes** in the Include Local System? field; otherwise type **No**.

Step 6. Press F4 to save the group definition.

A message appears that confirms that the group is defined.

Step 7. Press F5 (Resources) to access the Broadcast Services : Resource List (see Figure 10-3).

The resource list panel allows you to copy, browse, delete, and update resource definitions once they are part of a system group definition. The following steps describe how to add a new resource definition to your system group.

*Figure 10-3. Broadcast Services : Resource List*

```
PROD---------------- Broadcast Services : Resource List ---------------------
 Command ===>                                                Scroll ===> PAGE

                                          C=Copy S=Browse D=Delete U=Update
     Type     Name
     DOMAIN   PROD
     **END**

  F1=Help      F2=Split     F3=Exit      F4=Add       F5=Find      F6=Refresh
  F7=Backward  F8=Forward   F9=Swap
```

Step 8. Press F4 (Add) to add the required remote domains.

The Broadcast Services : Resource Definition panel is displayed.

*Figure 10-4. Broadcast Services : Resource Definition*

```
PROD---------------- Broadcast Services : Resource Definition ---------GROUP1
 Command ===>                                                   Function=ADD

 Resource Type ......               (APPCLINK, DOMAIN, LU)
 Resource Name ......

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

Step 9. Enter the type of resource in the Resource Type field.

The following types of resources are supported:

| | |
|---|---|
| APPCLINK | A predefined APPC Link name between two domains of up to 12 characters. |
| DOMAIN | A domain (as specified in the NMDID parameter) attached via INMC. The name can be up to 4 characters. |
| LU | A network resource name of up to 8 characters that identifies the required domain. |

**Note**

When adding Domain or LU type resources, ensure that the requisite DEFLINK commands have been issued to allow the domain to connect using APPC. See Chapter 7, *Using APPC to Communicate with Other Systems*, for more information on defining APPC links.

Step 10. Enter the name of the resource in the Resource Name field.

This name is used on the Broadcast Services : Send Menu when sending a broadcast to the system group.

Step 11. Press F3 (File) to save the changes and exit to the Resource List panel.

The resources that you added are listed. Multiple resources can be added by entering details for each resource and pressing F4 (Save).

# 11

# Customizing Printer Capabilities

This chapter provides information about how to set up and administer printing capabilities using the Print Services Manager (PSM). It covers the following topics:

**This chapter contains the following topics:**

- About Customizing PSM
- Accessing PSM
- Assigning Printer Definitions
- Specifying Paper Definitions
- Specifying Control Characters
- Specifying Default Printers
- Maintaining Your Printers
- Sending Print Requests to NET/MAIL
- Sending Print Requests to a Dataset

For information about using PSM, see the *Management Services User's Guide*.

## About Customizing PSM

You can customize PSM to provide the functionality needed to print from your system. It allows you to specify the format of a print request and on which printer it is to be printed. Print requests can be viewed on-line before or after printing and can be redirected to files rather than printers.

PSM provides the following features which can be customized to suit your requirements:

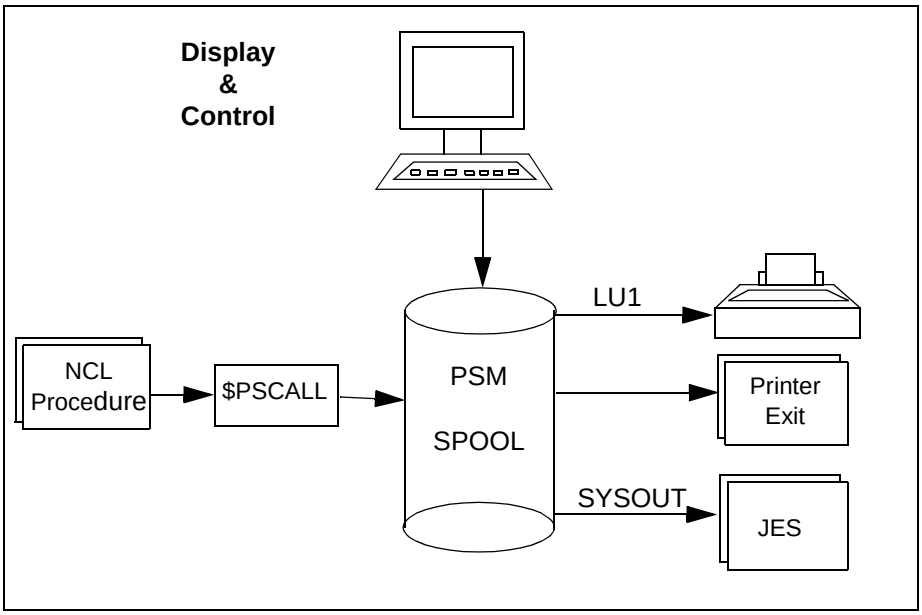
- Printer definition facilities
- Form definition maintenance
- Setup definition maintenance
- Default printer assignment maintenance
- Alias printer name definitions
- Banner page customization on output
- Spooled print request browsing, retention, and redirection to a different printer
- Integration with NCL-based components of Management Services

**Note**

References to JES in this chapter also apply to VOS3's JES3 and JES4 subsystems.

Figure 11-1 shows the different ways that PSM can be used to control printing requirements.

*Figure 11-1. Print Services Manager*




## Accessing PSM

The customizable functions of PSM are accessed from the PSM : Primary Menu (shortcut **/PSM**). For more information about the PSM : Primary Menu, see the *Management Services User's Guide*.

PSM can also be accessed by directly invoking the $PSCALL NCL procedure from OCS or an installation written NCL procedure. The PSM NCL interface is fully described in the *Network Control Language Reference*.

## Assigning Printer Definitions

A printer definition defines where, how, and on what paper output is to be printed. A printer definition is required for each printer at which output is to be printed.

The PSM : Printer Definition Menu is used to set up and administer printer definitions.

To access the printer definition menu, enter **P** at the Select Option ===> prompt on the PSM : Primary Menu.

The PSM : Printer Definition Menu is displayed.

*Figure 11-2. Printer Definition Menu*

```
PROD------------------- PSM : Printer Definition Menu -----------------$MENU10
 Select Option ===>

    A   - Add Printer Definition
    B   - Browse Printer Definition
    U   - Update Printer Definition
    D   - Delete Printer Definition
    C   - Copy Printer Definition
    L   - List Printer Definitions
    X   - Exit

 Printer Name ...  ____________    ( Required B U D C  Optional A L )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

For details of the field and options available on the PSM : Print Definition Menu, press F1 (Help).

## Adding Printer Definitions

The PSM : Printer Definition panel is displayed when you add, copy, update, or browse a printer definition. This section provides information on how to create a new printer definition, and describes the fields provided on the PSM printer definition panel.

Step 1. Enter **A** at the Select Option ===> prompt on the PSM : Printer Definition Menu.

The PSM : Printer Definition panel is displayed (see Figure 11-3).

Step 2. Type the required information in the fields and press F3 (File) to save the printer definition.

*Figure 11-3. Printer Definition Panel*

```
PROD---------------------- PSM : Printer Definition ----------------Page 1 of 1
 Command ===>                                                     Function=Add

 Printer Name .....  ___________
 Type .............  _____ (JES, VTAM, ALIAS, EXIT)

 Description ......  ______________________________________________
 Lower Case .......  YES          (YES or NO)
 Line Limit .......  0_____       (Range 0 to 999999)
 Form Name .......+  ___________
 ALIAS Printer
   Real Name .....+  ___________ (Real printer name)
 JES Printer
   Destination ....  __________________ (destid.userid)
   Output Class ...  F__          (One alphanumeric character)
 VTAM Printer
   LU Name ........  ________
   Logmode ........  ________
 EXIT
   Exit Name ......  ________
  Exit Data ...... ____________________________________________________________
                   ____________________________________________________________
  F1=Help      F2=Split     F3=File      F4=Save
                         F9=Swap                              F12=Cancel
```

For details of the information displayed on the PSM : Printer Definition panel, press F1 (Help).

## Listing Printer Definitions

You can display a list of all the printer definitions defined for your system. This list allows you to browse and perform maintenance on the listed definitions.

To access this list, enter **L** at the Select Option ===> prompt on the PSM : Printer Definition Menu.

The PSM : Printer Definition List is displayed (see Figure 11-4).

*Figure 11-4. Printer Definition List Panel*

```
PROD-------------------- PSM : Printer Definition List ----------------------
Command ===>                                                Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
   Name         Type   Description                                     File ID
   FILEMONTHLY  EXIT   Extract for monthly problem report             MODSUSR
   FILEWEEKLY   EXIT   Extract file for weekly problem report         MODSUSR
   MACHINE      ALIAS  Alias for machine room printer                 MODSUSR
   PRT001       JES    Network operations JES printer                 MODSUSR
   PRT002       VTAM   Machine room printer                           MODSUSR
   PRT003       JES    Tokyo office printer                           MODSUSR
   **END**

 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

You can display additional information about each printer definition in the selection list. To access this information, press F11 (Right). There are three additional panels.

For details of the information displayed and actions available on the PSM : Printer Definition List panel, press F1 (Help).

## Specifying Paper Definitions

A form definition is required for each type of paper on which output is to be printed. The form definition menu is used to set up and administer these paper definitions.

To access the form definition menu, enter **F** at the Select Option ===> prompt on the PSM : Primary Menu (shortcut **/PSM**).

The PSM : Form Definition Menu is displayed (see Figure 11-5).

*Figure 11-5. Form Definition Menu*

```
 PROD---------------------- PSM : Form Definition Menu -----------------$MENU20
  Select Option ===>

     A   - Add Form Definition
     B   - Browse Form Definition
     U   - Update Form Definition
     D   - Delete Form Definition
     C   - Copy Form Definition
     L   - List Form Definitions
     X   - Exit

  Form Name ...  ____________    ( Required B U D C  Optional A L )









F1=Help      F2=Split     F3=Exit      F4=Return
                            F9=Swap
```

For details of the field and options available on the PSM : Form Definition Menu, press F1 (Help).

## Adding Paper Definitions

The PSM : Form Definition panel is displayed when you add, copy, update, or browse a form definition. This section provides information on how to create a new form definition, and describes the fields provided on the PSM : Form Definition panel.

Step 1. Enter **A** at the Select Option ===> prompt on the PSM : Form Definition Menu.

The PSM : Form Definition panel is displayed (see Figure 11-6).

Step 2. Type the required information in the fields and press F3 (File) to save the form definition.

*Figure 11-6. Form Definition Panel*

```
PROD------------------------ PSM : Form Definition ---------------Page 1 of 1
Command ===>                                                      Function=Add

Form Name ..........  ___________


Brief Description ..  ____________________________________________
Full Description ...  ____________________________________________________________
                      ____________________________________________________________
                      ____________________________________________________________
Lines per Page .....  60_          (Range 0 to 999)
Columns per Page ...  132          (Range 1 to 256)
Banner Page ........  *NONE___     (*DEFAULT, *NONE or NCL procedure name)
Setup Name ........+  ___________

JES Printer
  JES Form Name ....  ____
  JES FCB Name .....  ____
  JES UCS Code .....  ____
  JES PGM Name .....  ________



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

For details of the fields on the PSM : Form Definition panel, press F1 (Help).

## Listing Paper Definitions

You can display a list of all the form definitions defined for your system and then browse and perform maintenance on them.

To access this form definition list, enter **L** at the Select Option ===> prompt on the PSM : Form Definition Menu.

The PSM : Form Definition List is displayed (see Figure 11-7).

*Figure 11-7. Form Definition List Panel 1*

```
PROD---------------------- PSM : Form Definition List ------------------------
Command ===>                                                   Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
    Name           Description                                        File ID
    FORM#10        10 Lines Per Page                                  MODSUSR
    FORM#20        20 Lines Per Page                                  MODSUSR
    FORM#60        60 Lines Per Page                                  MODSUSR
    FORM#66        66 Lines Per Page                                  MODSUSR
    LASER          Form definition for laser printer                  MODSUSR
    **END**

 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

To display additional information about each form definition in the selection list press F11 (Right). There are two additional panels.

For details of the fields and actions available on the PSM : Form Definition List, press F1 (Help).

## Specifying Control Characters

Control characters are sent to a printer before and/or after output is printed. They are defined in setup definitions. The setup definition menu allows you to:

- Add setup definitions
- Browse setup definitions
- Copy setup definitions
- Update setup definitions
- Confirm deletion of a setup definition

### Displaying the Setup Definition Menu

Enter **S** at the Select Option ===> prompt on the PSM : Primary Menu (shortcut **/PSM**).

The PSM : Setup Definition Menu is displayed, as shown in Figure 11-8.

*Figure 11-8. Setup Definition Menu*

```
SOLVPROD------------------ PSM : Setup Definition Menu ---------------$MENU030
 Select Option ===>

    A   - Add Setup Definition
    B   - Browse Setup Definition
    U   - Update Setup Definition
    D   - Delete Setup Definition
    C   - Copy Setup Definition
    L   - List Setup Definitions
    X   - Exit

 Setup Name ...  ____________    ( Required B U D C  Optional A L )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

For details of the field and options available on the PSM : Setup Definition Menu, press F1 (Help).

## Adding Control Characters

Step 1. Enter **A** at the Select Option ===> prompt on the PSM : Setup Definition Menu.

The PSM : Setup Definition panel is displayed (see Figure 11-9).

Step 2. To access the second panel of the setup definition, press F8 (Forward).

The second panel is shown.

Step 3. Type the required information in the fields and press F3 (File) to save the setup definition.

*Figure 11-9. Setup Definition Panel 1*

```
PROD----------------------- PSM : Setup Definition ---------------Page 1 of 2
Command ===>                                                     Function=Add

Setup Name ......... ____________


Brief Description .. ______________________________________________
Full Description ... ______________________________________________________________
                     ______________________________________________________________
                     ______________________________________________________________

Initialization Data in Hexadecimal
   ______________________________________________________________________________
   ______________________________________________________________________________
   ______________________________________________________________________________
   ______________________________________________________________________________
   ______________________________________________________________________________
   ______________________________________________________________________________
   ______________________________________________________________________________
   ______________________________________________________________________________
   ______________________________________________________________________________
   ______________________________________________________________________________

F1=Help      F2=Split     F3=File      F4=Save
              F8=Forward   F9=Swap                                F12=Cancel
```

For details of the fields on the PSM : Setup Definition panel, press F1 (Help).

## Listing Control Characters

You can display a list of all the setup definitions defined for your system. This list allows you to browse and perform maintenance on the listed definitions.

Enter **L** at the Select Option ===> prompt on the PSM : Setup Definition Menu. The PSM : Setup Definition List is displayed (see Figure 11-10).

*Figure 11-10. Setup Definition List Panel 1*

```
PROD---------------------- PSM : Setup Definition List ---------------------
Command ===>                                                  Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
    Name           Description                                        File ID
    SETUPL         Set laser printer to landscape                     MODSUSR
    SETUP1         Setup for type 1 laser printer                     MODSUSR
    SETUP2         Setup for type 2 laser printer                     MODSUSR
    SETUP3         Setup for type 3 laser printer                     MODSUSR
    **END**

 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

Additional information is displayed about each setup definition in the selection list. To access this information, press F11 (Right). There is one additional panel.

For details of the information displayed and actions available on the PSM : Setup Definition List panel, press F1 (Help).

## Specifying Default Printers

Each user ID in your system can be assigned a default printer. Default printer assignments allow you to define the printer to which output is sent whenever a user ID does not specify a printer.

The default printer assignment is defined using the PSM : Default Printer Assignment Menu.

### Displaying the Default Printer Assignment Menu

Enter **D** at the Select Option ===> prompt on the PSM : Primary Menu (shortcut **/PSM**).

The PSM : Default Printer Assignment Menu is displayed (see Figure 11-11).

*Figure 11-11. Default Printer Assignment Menu*

```
PROD---------------- PSM : Default Printer Assignment Menu -------------$PS040
 Select Option ===>

    A   - Add Default Printer Assignment
    B   - Browse Default Printer Assignment
    U   - Update Default Printer Assignment
    D   - Delete Default Printer Assignment
    C   - Copy Default Printer Assignment
    L   - List Default Printer Assignments
    X   - Exit

 Userid ...  ________    ( Required B U D C  Optional A L )









 F1=Help      F2=Split     F3=Exit      F4=Return
                           F9=Swap
```

For details of the field and options available on the PSM : Setup Definition Menu, press F1 (Help).

## Adding a Default Printer for a User ID

The PSM : Default Printer Assignment is used to add, browse, update and confirm deletion of a default printer assignment. This section provides information on how to create a new default printer definition.

Step 1. Enter **A** at the Select Option ===> prompt on the PSM : Default Printer Assignment Menu.

The PSM : Default Printer Assignment is displayed (see Figure 11-12).

*Figure 11-12. Default Printer Assignment Panel*

```
PROD------------------ PSM : Default Printer Assignment ----------Page 1 of 1
Command ===>                                                     Function=Add

Userid ..........  ________
Printer Name ...+  ____________



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

For details of the information displayed on the PSM : Default Printer Assignment panel, press F1 (Help).

Step 2. Type the required information in the fields and press F3 (File) to save the default printer assignment.

## Listing Default Printers

You can display a list of all the default printer assignments defined for each user ID. This list allows you to browse and perform maintenance on the listed definitions.

Step 1. Enter **L** at the Select Option ===> prompt on the PSM : Default Printer Assignment Menu.

The PSM : Default Printer Assignment List is displayed (see Figure 11-13).

Step 2. To access additional information about each default printer assignment in the selection list, press F11 (Right). There is one additional panel.

*Figure 11-13. Default Printer Assignment List Panel 1*

```
PROD------------------ PSM : Default Printer Assignment List ----------------
Command ===>                                                  Scroll ===> PAGE

                                          S/B=Browse U=Update D=Delete C=Copy
    Userid     Printer                                                File ID
    USER01     PRT001                                                 MODSUSR
    USER02     PRT002                                                 MODSUSR
    USER03     PRT003                                                 MODSUSR
    USER04     PRT004                                                 MODSUSR
    USER05     PRT005                                                 MODSUSR
    **END**

 F1=Help      F2=Split     F3=Exit      F4=Return    F5=Find      F6=Refresh
 F7=Backward  F8=Forward   F9=Swap                  F11=Right
```

For details of the information displayed and actions available on the PSM : Default Printer Assignment List panel, press F1 (Help).

## Maintaining Your Printers

The administration facility of PSM allows you to maintain default information for your printers. It also allows you to maintain the print spool.

### Accessing the Administrative Functions Menu

Enter **A** at the Select Option ===> prompt on the PSM : Primary Menu (shortcut **/PSM**).

The PSM : Administration Menu is displayed (see Figure 11-14).

*Figure 11-14. PSM : Administration Menu*

```
PROD---------------------- PSM : Administration Menu -----------------$PS050
Select Option ===>

   BD  - Browse Defaults Definition
   UD  - Update Defaults Definition
   CS  - Clear the Spool
   X   - Exit



F1=Help      F2=Split     F3=Exit      F4=Return
                          F9=Swap
```

For details of the options available on the PSM : Administration Menu, press F1 (Help).

### Maintaining Printer Defaults

The information provided in the defaults definition is used as the default values for certain fields on the Printer and Form Definition panels when using the Add function. These values are also used as defaults for certain keywords on the open call to the PSM NCL interface procedure $PSCALL.

The Defaults Definition panel is displayed when browsing and updating the defaults definition.

## Updating the Defaults Definition

Enter **UD** at the Select Option ===> prompt on the PSM : Administration Menu.

The PSM : Defaults Definition is displayed (see Figure 11-15).

*Figure 11-15. PSM : Defaults Definition Panel*

```
PROD---------------------- PSM : Defaults Definition --------------Page 1 of 1
Command ===>                                                     Function=Browse

Printer Defaults                                        Last update by USER01
 Lower Case? ........  YES       (YES or NO)          Last update on 13-APR-92
 Line Limit .........  0         (Range 0 to 999999  Last update at 16.20.33
 Form Name ..........
 Output Class .......  F         (One alphanumeric character)
 Logmode ............


Form Defaults
 Lines per Page .....  60        (Range 0 to 999)
 Columns per Page ...  132       (Range 1 to 256)
 Banner Page ........  *NONE     (*DEFAULT, *NONE or NCL procedure name)









 F1=Help      F2=Split     F3=Exit
                           F9=Swap
```

For details of the fields on the PSM : Administration Menu, press F1 (Help).

## Clearing the Printer Spool

Print requests are retained on the print spool if an error occurs during printing or if HELD is specified on the PSM : Print Request panel. The PSM clear spool panel is used to clear print requests from the print queue. This function is only available to authorized users.

To clear entries in the print spool, perform the following steps:

Step 1. Enter **CS** at the Select Option ===> prompt on the PSM : Administration Menu.

The PSM : Clear Spool panel, is displayed (see Figure 11-16).

*Figure 11-16. Clear Spool Panel*

```
PROD------------------------- PSM : Clear Spool -----------------Page 1 of 1
Command ===>


Date ... __________ (Requests created on or before this date will be deleted)



F1=Help      F2=Split     F3=Exit                                F6=Action
                          F9=Swap
```

Step 2. Specify a date in the Date field.

All requests added to the spool on or before this date are deleted. For information on date formats that can be specified, see the following section, Date Formats.

Step 3. Press F6 (Action) to complete the deletion.

## Date Formats

The date on the clear spool panel can be entered in any one of the following formats:

- DD-MMM-CCYY
- DD/MM/YY or MM/DD/YY

  Dependent on the language code of the user
- DD

  The month and year are assumed to be the current month and year
- DDMM or MMDD

  Dependent on the language code of the user
- DDMMYY or MMDDYY

  Dependent on the language code of the user
- DDD

  The year is assumed to be the current year
- YYDDD or YY.DDD

Date formats consist of the following abbreviations:

| | |
|---|---|
| DD | the day of the month |
| DDD | the day of the year |
| MMM | the month, for example, JAN, FEB, MAR |
| MM | the month, for example, 01, 02, 03 |
| YY | the year excluding the century |
| CCYY | the year |

The following values can also be entered into the date field to retrieve the current date or a date based on the current date:

| | |
|---|---|
| = | retrieves the current system date |
| +*nnn* | the number of days after the current date |
| -*nnn* | the number of days before the current date |
| *dow* | the day of the current week—valid values are MON TUE WED THU FRI SAT SUN |

| | |
|---|---|
| *+dow* | the day of the following week |
| *-dow* | the day of the previous week |

### Distributed Banner Exit Procedure

The distributed banner exit, which is named $PSBA80X, prints a banner page before the output for a print request. Information printed includes the requesting user's user ID, name, location, phone number, print request number, and the date and time.

## Sending Print Requests to NET/MAIL

You can send a print request to NET/MAIL using a printer exit. The exit can be used to send a print request to the following:

- The NET/MAIL ID of the output requestor
- A specific NET/MAIL ID
- A NET/MAIL mailing list (either private or public)

Options can also be specified which enable the format of the NET/MAIL items to be tailored in their appearance.

Two printer exit procedures are distributed with Management Services to interface with different versions of NET/MAIL. The procedure names and versions of the two printer exits are as follows:

- $PSML78X—PSM to NET/MAIL V2.2 and earlier
- $PSML79X—PSM to NET/MAIL V2.2.2 and later

The version 2.2.2 and later printer exit can be operated from a system other than that on which NET/MAIL is available. That is, PSM on system X can print output which will create mail items on system Y.

### Prerequisites for Using the Printer Exit Procedure

If you want to use the remote access facility, you require Management Services V3.0 or later for both the PSM and the NET/MAIL systems. NET/MAIL PTF Z57065 is required for NET/MAIL V2.2.2.

## Who Does the Mail Come From?

The exit creates mail items as if they were sent by the user that requested the output to be generated. For instance, the user who directs a Report Writer report to a PSM printer (which is defined as an exit) becomes the *sender* of the resulting mail item(s).

## Output Segmentation

The maximum number of records output by PSM can be limited by specifying the *Maximum Lines* in the printer definition. If this count is exceeded, the print request is marked in error by PSM. However, if the Maximum Lines is defaulted to 0, or is set to a value greater than the maximum possible size of a NET/MAIL mail item, then it is possible that the exit will receive more output than it can post in a NET/MAIL item.

In this case the exit segments the output into multiple NET/MAIL items. Data is not lost but simply continued in one or more subsequent mail items. Currently the maximum number of text lines available in a NET/MAIL V2.2.2 mail item is 9995 lines.

**Note**

This is after any formatting performed by the exit.

## Using the Remote Access Facility

The remote access facility allows the exit to access NET/MAIL on a remote system. It requires an APPC connection between the PSM system and the NET/MAIL system. This requires that appropriate definitions be made in both the systems. The DEFOPSET and DEFLINK commands are used to define the configuration required to enable an APPC conversation to be established.

Following are two examples of definitions for APPC connectivity:

```
DEFOPSET OPSET=PSM MAP=YES LOG=YES SECURITY=USER
```

```
DEFLINK TYPE=APPC LUNAME=luname LINK=linkname OPSET=PSM
```

APPC security must be satisfied in order for the remote access facility to function correctly. This is discussed further in the section titled *$PSML78X and $PSML79X Parameters*.

For more information about the DEFOPSET and DEFLINK commands, see the *Management Services Command Reference*.

## Output Formatting Considerations

The actual format of the NET/MAIL item generated by the exit is dependent on the interaction of many variables. For example:

- The SKIP, PAGE, and USCORE parameters passed to the exit in the exit data
- The FORM definition used by PSM when passing the output to the exit
- The usage of the SKIP and USCORE options by the procedure generating the output

In order to generate NET/MAIL items of the expected format, it is important to understand the interaction of these parameters.

## $PSML78X and $PSML79X Parameters

Parameters which control the operation of the exit are defined in the *Exit Data* portion of the printer definition. Procedures which pass data to PSM for printing have the ability to override the exit data specified in the PSM printer definition.

The exits use the parameters contained within the exit data to do the following:

- Locate the NET/MAIL system
- Determine the NET/MAIL target ID and mail item options
- Determine how to format the data passed by PSM before passing it to NET/MAIL

The $PSML78X and $PSML79X parameters, which are coded as keyword parameters, are identical for both versions of the exit with the exception of the CLASS parameter which is not supported for NET/MAIL V2.2 and earlier. The parameters are as follows:

```
[ MAILID={ mailuser | *maillist } ]
[ CLASS=xx ]
[ VIEWONLY={ YES | NO } ]
[ PRIVATE={ YES | NO } ]
[ SUBJECT= subject ]
[ LU=luname|LINK=linkname|DOMAIN=domain]
[ PROFILE= profile ]
[ SKIP={ ON | OFF | FWD } ]
[ PAGE={ ON | OFF | LINE } ]
[ USCORE={ ON | OFF } ]
```

**MAILID={ *mailuser* | **maillist* }**

Specifies the NET/MAIL destination ID for the output data. If this parameter is omitted, the user ID of the user who requested the generation of the output is used as the MAILID. An exit definition which does not contain a MAILID is a generic definition which delivers the PSM output to the requestor's own NET/MAIL ID.

A NET/MAIL mailing list can be specified by prefixing the name of the list with an asterisk (*). The mailing list can be a *public* or *private* list. If the list is a private list it must belong to the mail sender, that is, the user who instigated the generation of the PSM output.

**CLASS=*xx***

This parameter is valid only for NET/MAIL systems V2.2.2 and later. It is ignored by the exit for NET/MAIL V2.2.

This parameter over-rides the default NET/MAIL mail class. Mail classes are defined within NET/MAIL and comprise a 2 digit number. The class specification is checked at the destination by NET/MAIL itself, not by the exit. The default is determined by the target NET/MAIL system.

**VIEWONLY={ YES | NO }**

Specifies whether NET/MAIL is to treat this as a VIEWONLY mail item. VIEWONLY items cannot be saved to folders and cannot be forwarded. The default is NO.

**PRIVATE={ YES | NO }**

Specifies whether NET/MAIL is to treat this as a PRIVATE mail item. PRIVATE mail items cannot be viewed by surrogate users. The default is NO.

**SUBJECT= *subject***

Specifies subject text which appears as the SUBJECT of the NET/MAIL mail item. The subject text replaces the default text which is automatically generated by the exit. If the subject text contains imbedded blanks it must be quoted. The default subject text generated by the exit is:

`Print Services Manager request` *`xxxx`* `from` *`yyyyyyyy`*`.`

where *xxxx* is the PSM request number and *yyyyyyyy* is the system ID where PSM is executing.

If more than one mail item is created for a given PSM request number, mail items subsequent to the first have the text

`(item` *`z`*`)`

appended to the subject, where *z* is the mail item sequence number (for example, 2, 3, 4).

**LU=*luname* | LINK=*linkname* | DOMAIN=*domain***

Specifies the APPC conversation destination ID used to access NET/MAIL on a remote system. If omitted, the exit attempts to access NET/MAIL on the local system.

If the DOMAIN option is used, an INMC link to the destination system must already be active before any print request can be successfully processed. Both the PSM and the NET/MAIL systems must be Management Services V3.0 or later systems for the remote access facility to function.

**PROFILE=*profile***

This keyword is only valid for remote NET/MAIL access. It specifies the user profile under which the NET/MAIL process is executed on the NET/MAIL system. The default is the USERID of the BLOG region where the exit is executing.

It is suggested that the BLOG user region of the NET/MAIL system be used for this purpose and that APPC user *keys* and *locks* be set up accordingly to provide the required security access. See the *Network Control Language User's Guide* for information about setting up APPC keys and locks.

**SKIP={ ON | OFF | FWD }**

Determines how the exit will action line skip options passed by PSM with the output data. Line skip options are specified by the procedure which generated the PSM output. See the section on $PSCALL OPT=PUT in the *Network Control Language Reference* for more information.

If ON is specified, the exit honors skip options of 0, 1, 2 and 3. If OFF is specified, all skip options (including SKIP=0) are treated as if they were SKIP=1. That is, each PSM output text line appears immediately following the previous line regardless of the SKIP options used.

Specifying FWD indicates that only forward skip options (1, 2 and 3) are honored, SKIP=0 lines are discarded.

SKIP=0 lines is normally used to underline the text, or to overlay the previous text with either the same text (bolding) or with new text (obliterating). If SKIP=ON is specified and a SKIP=0 operation is encountered, then an attempt is made to emulate the original intent of the operation. The first character of the SKIP=0 line is checked and if a minus or underscore then the line is added as if SKIP=1 were specified, as the likely intention was to underline the previous line of text. This operation adds a line of text to the NET/MAIL item. If the first character was not a minus or underscore character then the SKIP=0 data is overlaid onto the previous line of text. This does not add a line.

**PAGE={ ON | <u>OFF</u> | LINE }**

Specifies how the exit actions page breaks specified by the SKIP=NEWPAGE option on PSM output data.

If PAGE=OFF is specified, the SKIP=NEWPAGE is treated as a SKIP=1 and no intervening lines are added to the NET/MAIL item.

If PAGE=LINE is specified, SKIP=NEWPAGE is treated as a SKIP=2, causing one blank line to separate the previous text line from the current text line.

Specifying PAGE=ON causes the NET/MAIL interface to generate a page separation banner. The banner comprises two blank lines followed by the centered text:

```
-----page----
```

followed by two blank lines.

**USCORE={ <u>ON</u> | OFF }**

Determines how the exit actions the USCORE=1 and USCORE=2 options on PSM output data.

If USCORE=ON is specified, the exit emulates underscoring of the text line. The USCORE emulation causes a line of text to be added to the NET/MAIL item.

If USCORE=OFF is specified, the exit bypasses USCORE emulation and therefore maintains the original number of output lines.

**Note**

The PSM print option of BOLD is ignored by the exit.

## Sample Printer Exit Definitions

The following section provides three printer exit definition samples:

### Sample 1

This is an example of a definition which sends the PSM output to the NET/MAIL ID of whoever instigated the output. The NET/MAIL system is version 2.2 and is available in the local system. The FORM0 form specifies no page breaks.

| | |
|---|---|
| Printer Name: | NETMAILSELF |
| Type: | EXIT |
| Description: | PSM O/P goes to your own NET/MAIL ID on this system. |
| Lower Case: | YES |
| Line Limit: | 0 |
| Form Name: | FORM0 |
| Exit Name: | $PSML78X |
| Exit Data: | |

### Sample 2

This is an example of a definition which sends the PSM output to a NET/MAIL mailing list. The NET/MAIL system is version 2.2.2 and is remote to this PSM. The remote NET/MAIL procedure is run in the BLOG user region (user ID of PRODBLOG) of that system.

| | |
|---|---|
| Printer Name: | EMAILACCNTS |
| Type: | EXIT |
| Description: | Mail to entire accounts department. |
| Lower Case: | YES |
| Line Limit: | 0 |
| Form Name: | FORM0 |
| Exit Name: | $PSML79X |
| Exit Data: | MAILID=*ACCOUNTS LINK=PRODMAIL PROFILE=PRODBLOG |

### Sample 3

In this example a form name of FORM37 defines a page length of 37 lines, being the text display area for a NET/MAIL (version 2.2.2) item on a 43 line 3270 screen. The exit parameters for mail item formatting are set so as to not interfere with the number of output lines so that the page breaks generated by PSM match the screen display area size.

| | |
|---|---|
| Printer Name: | EXITTEST |
| Type: | EXIT |
| Description: | PSM NET/MAIL Interface testing. |
| Lower Case: | YES |
| Line Limit: | 0 |
| Form Name: | FORM37 |
| Exit Name: | $PSML79X |
| Exit Data: | MAILID=FRED PAGE=OFF SKIP=FWD USCORE=OFF |

## Sending Print Requests to a Dataset

Two printer exit procedures are distributed with Management Services. Each will write the output for a print request to a dataset. The procedure $PSDS81X can be customized to specific site requirements. The procedure $PSDS81Z offers the same functionality with improved performance. It cannot be customized. The target datasets for both procedures can be either sequential or partitioned.

Parameters which control the operation of the exit are defined in the *Exit Data* portion of the printer definition. Procedures which pass data to PSM for printing have the ability to override the exit data specified in the PSM printer definition.

The procedures use the parameters contained within the exit data to do the following:

- Determine the target dataset
- Determine how to process a data line with a skip amount of zero
- Set the length of the lines to be printed

### How the Procedures Process a Print Request

The procedures read each line of print data and write it directly to the nominated dataset. Each print line is analyzed according to skip control before processing. This continues until all lines of data for the print request have been received from PSM and written to the nominated dataset.

### $PSDS81X and $PSDS81Z Parameters

The $PSDS81X and $PSDS81Z parameters, which are coded as keyword parameters, are as follows:

```
  DSN=datasetname
[ DISP={ SHR | OLD | MOD } ]
[ LRECL={ n | 80 } ]
[ SKIP0={ NEWLINE | DISCARD | DESTRUCTIVE |
          NONDESTRUCTIVE } ]
```

**DSN=*datasetname***

Specifies the target dataset name. If the dataset is partitioned, the member name must be included or the dataset will be corrupted.

**DISP={ SHR | OLD | MOD }**

Specifies the disposition of the output dataset. SHR specifies shared use of the dataset. OLD specifies exclusive use of the dataset. MOD causes the output to be appended in the file. The default is SHR.

**LRECL={ *n* | <u>80</u> }**

Specifies the output record length. The valid range for LRECL is 1 to 250. The default is 80.

**SKIP0={ <u>NEWLINE</u> | DISCARD | DESTRUCTIVE | NONDESTRUCTIVE }**

Specifies how to process a data line with a skip amount of zero. NEWLINE creates a new line of data. DISCARD causes the line of data to be discarded. DESTRUCTIVE causes the data to replace the existing data line. NONDESTRUCTIVE causes the data to be overlaid on the existing data line, but only where blanks were present on the existing data line. No existing non-blank characters are modified. The default is NEWLINE.

**Note**

The PSM print options NEWPAGE and USCORE are ignored by the procedures

## Sample Printer Exit Definition

This example directs the output for a PSM print request, assigned to the printer named DSEXIT, to the member TEST1 in the dataset PROD.PSM.DATA. The record length of this dataset is 80. Overlay lines in the data are removed.

| | |
|---|---|
| Printer Name: | DSEXIT |
| Type: | EXIT |
| Description: | Print to a dataset |
| Lower Case: | YES |
| Line Limit: | 0 |
| Form Name: | FORM0 |
| Exit Name: | $PSDS81Z |
| Exit Data: | DSN=PROD.PSM.DATA(TEST1) LRECL=80 SKIP0=DISCARD |

**Note**

Previous references to parameters WKVOL, CYL, and LIST in the Exit data are no longer required. You must remove them from the printer definition prior to using $PSDS81Z or $PSDS81X, or the print request will fail.

# 12

# Implementing Activity Logs

This chapter provides information about activity logs and how to implement them.

**This chapter contains the following topics:**


- About the Activity Logs
- About the Online Activity Log
- Implementing Online Activity Logging
- Browsing Online Logs
- Swapping the Online Log
- Dealing with Errors on the Online Log
- Recording Additional Information on the Online Log
- Using a Log Exit for the Online Log
- Replacing Your Online Logging Procedure
- About the Hardcopy Activity Log
- Swapping the Hardcopy Log
- Wrapping the Hardcopy Log Datasets (OS/VS)
- Cross Referencing Hardcopy Logs
- Dealing with I/O Errors on the Hardcopy Log (OS/VS)
- Recording Additional Information on the Hardcopy Log
- Writing to the System Log (OS/VS)

## About the Activity Logs

Management Services (MS) provides a logging facility that records all of the activity in your MS region. This is called activity logging. The activity logs can be used to aid in problem determination.

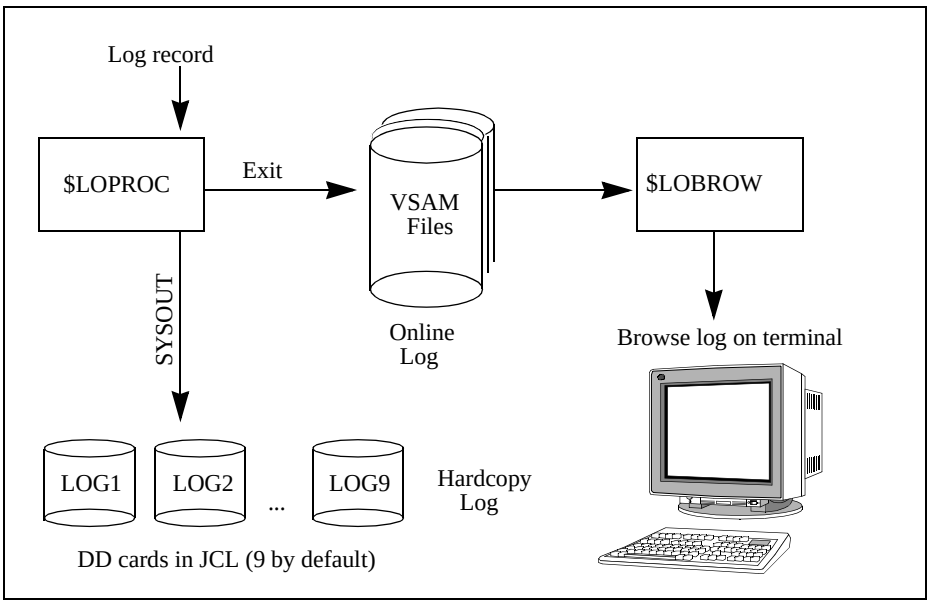
There are two separate activity logs:

- Online
- Hardcopy

By default, activity logs contain the following information:

- All commands entered
- All responses to commands entered
- Any unsolicited messages received from VTAM or the operating system, provided the related interfaces are available
- All messages explicitly written to the log by NCL procedures

*Figure 12-1. Logging Facilities*



## About the Online Activity Log

The online activity log is supplied by the distributed procedure $LOPROC. The $LOPROC procedure writes log data to VSAM files (three by default). The VSAM files are accessed by a second procedure, $LOBROW, which allows online browsing of the log. Figure 12-1 shows how this process works.

**Note**

$LOPROC and $LOBROW are the default procedure names. You can change these by using the LOGFILES parameter group in ICS.

## Implementing Online Activity Logging

During initialization, the region allocates, by default, three VSAM log files for online logging. However, you can allocate up to seven files.

**Note**

The log file IDs are of the form NMLOG*nn* and the dataset names are of the form *dsnpref.rname*.NMLOG*nn*. (*dsnpref* is the dataset prefix used during product installation and *rname* is the name of the region.)

### Using Additional Log Files

If you want to make more than three files available to the region, define the new VSAM files and then customize the LOGFILES parameter group by defining additional logging datasets.

To customize the LOGFILES parameter group, do this:

Step 1. Enter **/ICS** at the ===> prompt. The ICS : Customization Parameters list is displayed.

Step 2. Enter **U** beside the LOGFILES parameter group. The Initialization Parameters panel for the LOGFILES parameter group is displayed.

*Figure 12-2. The LOGFILES Parameter Group*

```
PROD------------------ ICS : Initialization Parameters -------------Page 1 of 3
Command ===>                                                  Function=Update

.- LOGFILES - Log File Specifications ----------------------------------------.
|                                                                              |
| Are Activity Logs to Wrap? ... YES       (Yes or No)                         |
| Log Operating System Messages? NO        Log Commands? ... NO   (Yes or No) |
| Log Procedure Name ........... $LOPROC   Log Exit Name ...                   |
| Log Viewing Procedure Name ... $LOBROW   (default is $LOBROW)                |
| Log View FIND Line Limit ..... 2000                                          |
| Log File ID 1 ................ NMLOG01   (default is NMLOG01)               |
|   Log File Dataset Name 1 .... ALSO00.PROD1.NMLOG01_________________________ |
|                                (default is AUDE0.PROD1.NMLOG01)              |
|   File Disposition 1 ........+ SHR                                           |
|   Log File VSAM Options 1 ...+ LSR SIS DEFER________________________________ |
|                                (default is LSR SIS DEFER)                    |
'------------------------------------------------------------------------------'
```

Step 3. From the displayed Initialization Parameters panel, press F8 (Forward) to display the next page.

Step 4. Complete the fields for each file you want to make available. Figure 12-3 shows an example.

*Figure 12-3. Allocating an Activity Log File*

```
PROD------------------ ICS : Initialization Parameters -------------Page 2 of 3
Command ===>                                                  Function=Update

.- LOGFILES - Log File Specifications ----------------------------------------.
|                                                                              |
| Log File ID 2 ................ NMLOG02                                     |
|   Log File Dataset Name 2 .... ALSO0.PROD1.NMLOG02                           |
|   File Disposition 2 ........+ SHR                                           |
|   Log File VSAM Options 2 ...+ LSR SIS DEFER________________________________ |
|                                                                              |
| Log File ID 3 ................ NMLOG03                                     |
|   Log File Dataset Name 3 .... ALSO00.PROD1.NMLOG03                          |
|   File Disposition 3 ........+ SHR                                           |
|   Log File VSAM Options 3 ...+ LSR SIS DEFER________________________________ |
|                                                                              |
| Log File ID 4 ................ NMLOG04                                     |
|   Log File Dataset Name 4 .... ALSO00.PROD1.NMLOG04                          |
|   File Disposition 4 ........+ SHR                                           |
|   Log File VSAM Options 4 ...+ LSR SIS DEFER________________________________ |
```

Step 5. To allocate more files, press F8 (Forward) again.

Step 6. When you have specified the required number of log files, press F6 (Action) to allocate and open the files.

Step 7. When the parameter group has completed its actions, press F3 (File) to save the changed information.

For further information about using this panel, press F1 (Help).

## Browsing Online Logs

To access the online activity log, do any of the following:

- Enter **/LOG** at the ===> prompt on a panel
- Enter **$LOG** *linkname* at the ===> prompt on a panel
- Press F7 in OCS.

For information about browsing the online activity log, press F1 (Help) from the activity log panel.

The log browse facility displays all messages in the online activity log, allowing you to browse logically concatenated online files.

To assist in locating data in the file, the log browse facility supports many commands. It also makes use of the function keys to simplify operation.

The online activity log has the following capabilities:

- Forward and backward scroll
- Positioning by date or time, either absolute or relative
- Search forwards or backwards for text
- Repositioning by labels set within the log
- Execution of commands
- Full-screen help for all log-browse commands
- Various display formats
- Context-sensitive help
- Browsing the online activity log for a linked region
- Printing portions of the log

## Administering Online Activity Log Files

To administer the online activity log, enter **/LOADMIN** at the ===> prompt. The Activity Log : Administration menu is displayed.

*Figure 12-4. The Activity Log : Administration Menu*

```
PROD------------------- Activity Log : Administration ----------------/LOADMIN
Select Option ===>

   S   - Swap Active Activity Log NMLOG03                  LOGSWAP
   D   - List All Days Contained in Log Files              LOGDAYS
   F   - List All Physical Log Files                       LOGFILE
   X   - Exit

Domain Name ............+                                       (Optional D)



 F1=Help      F2=Split     F3=Exit      F4=Return
                          F9=Swap
```

For details of the options available on this menu, press F1 (Help).

## Swapping the Online Log

The online activity log automatically swaps to a fresh VSAM file when each file fills up.

You can manually swap your currently active VSAM if you wish to free a particular log file (for example, for backups).

**Caution**

Swapping the current VSAM log causes the $LOPROC procedure to write all subsequent activity log records to the next VSAM log. If this log was previously used, it is reset; so you can no longer browse the old records that it contained.

To swap the online activity log, do this:

Step 1. Enter **/LOGSWAP** at the ===> prompt. The Activity Log Services : Confirm Swap Log panel is displayed.

*Figure 12-5. Activity Log Services : Confirm Swap Log Panel*

```
PROD--------------- Activity Log Services : Confirm Swap Log --------------------
Command ===>

                                SWAP ACTIVITY LOG
.-------------------------------------------------------------------------------.
| This action will swap activity logging to the next defined VSAM log file.    |
|                                                                               |
| The new log will be emptied before swapping.                                  |
| If only one log file is available, it will be emptied and re-used.            |
|                                                                               |
| Any procedure that is currently accessing the log file may be flushed by      |
| this operation.                                                               |
.-------------------------------------------------------------------------------.
.
 Press the Action key to request the log swap, or the Cancel key to cancel it.
```

Step 2. Press F6 to request the log swap, or F12 to cancel your request.

## Dealing with Errors on the Online Log

If the $LOPROC procedure encounters a VSAM error when it is logging activity to an online log file, it automatically swaps to the next log file.

## Recording Additional Information on the Online Log

You can use the LOG command to record additional information on the log that is outside the scope of that recorded by default. For example, enter at the Command ===> prompt:

```
LOG PRODUCTION LINK TO LOS ANGELES DOWN FOR MAINTENANCE
```

This produces the following entry in the log for the time the command was issued:

```
LOG ENTRY : LOG PRODUCTION LINK TO LOS ANGELES DOWN FOR
MAINTENANCE
```

The message recorded can be up to 256 characters long. Additional messages can also be written to the log using the &WRITE LOG=YES NCL statement from NCL procedures.

## Using a Log Exit for the Online Log

You can create an NCL procedure to intercept, analyze, and react to the messages that are sent to the activity log.

Use the LOGFILES parameter group in ICS to specify the name of your exit.

The exit is executed every time a message is sent to the log. Using the exit to perform complex functions can degrade the performance of the region.

**Note**

Ensure that your log exit procedure is well tested before you put it into production.

### Variables Available to the Activity Log Exit

Figure 12-1 describes the variables that are available to the exit.

*Table 12-1.Activity Log Exit—Variables*

| Variable | Description |
|---|---|
| &#LO$RECORD | Contains the following record:<br>*time-generated user-id terminal-id message-text*<br><br>The text of the message starts at the 4th word of the record. For more information, see the &LOGREAD verb in the *Network Control Language Reference*.<br><br>You can change the contents of this variable. To suppress the message from the log, set the variable to NOLOG. |
| $LOG | A Mapped Data Object (MDO) that contains the message attributes; the MDO is mapped by the $MSG map.<br><br>You can use the &ASSIGN verb to query the MDO. For information about querying MDO components, see the *Network Control Language User's Guide*. |

## Enabling the Log Exit

To enable the exit, do this:

Step 1. Enter **/ICS** at the ===> prompt. The ICS : Customization Parameters list is displayed.

Step 2. Enter **U** beside the LOGFILES parameter group. The Initialization Parameters panel for the LOGFILES parameter group is displayed.

*Figure 12-6. The LOGFILES Parameter Group*

```
PROD------------------ ICS : Initialization Parameters -------------Page 1 of 3
Command ===>                                                  Function=Update

.- LOGFILES - Log File Specifications ----------------------------------------.
|                                                                              |
| Are Hardcopy Logs to Wrap? ... YES       (Yes or No)                         |
| Log Operating System Messages? NO        Log Commands? ... NO   (Yes or No) |
| Log Procedure Name ........... $LOPROC   Log Exit Name ...                   |
| Log Viewing Procedure Name ... $LOBROW   (default is $LOBROW)                |
| Log View FIND Line Limit ..... 2000                                          |
| Log File ID 1 ............... NMLOG01   (default is NMLOG01)                |
|   Log File Dataset Name 1 .... ALSO00.SOLV17.NMLOG01_______________________ |
|                                (default is AUDE0.DENM13.NMLOG01)             |
|   File Disposition 1 ........+ SHR                                           |
|   Log File VSAM Options 1 ...+ LSR SIS DEFER________________________________ |
|                                (default is LSR SIS DEFER)                    |
'------------------------------------------------------------------------------'
```

Step 3. Enter the name of your activity log exit in the Log Exit Name field.

Step 4. Press F6 to action the changes.

Step 5. Press F3 to file the changes.

# Replacing Your Online Logging Procedure

The default online logging procedure is $LOPROC. This procedure is designed to work in conjunction with the online browsing procedure $LOBROW.

You can replace the $LOPROC and $LOBROW procedures with your own customized NCL procedures. Alternatively, you can write a customized log browsing procedure to present the supplied data files (from $LOPROC) in your own format.

## Writing Your Own Log Browsing Procedure

To write your own customized NCL procedure to replace $LOBROW, use the &FILE OPEN statement with FORMAT=DELIMITED.

The physical file structure of the supplied log files (NMLOG01, NMLOG02, and NMLOG03) is as follows:

**Key format**
YYYYMMDDHHMMSSHS*nnnn*
where *nnnn*=1000 + (reset every 100th of a second)
and key length=20 bytes

**Record contents**
Delimited fields:

| | |
|---|---|
| ORIGIN | Terminal name |
| REGION | User ID |
| TEXT | Message text to be displayed in the activity log |
| MSGATTR | 2-byte color/highlight indicator.<br>Color can be: R=red, Y=yellow, W=white, B=blue, G=green, T=turquoise, or P=pink<br>Highlighting can be: R=reverse, B=blink, U=uscore, or N=none |
| ORIGTIM | Remote domain time |
| ORIGDMN | Originating domain name |
| ORIGSRC | Remote terminal ID |

For further information, see the following:

- The description of the &FILE OPEN verb in the *Network Control Language Reference*
- The *NCL File Processing* chapter in the *Network Control Language User's Guide*

### Writing Your Own Logging and Browsing Procedures

You can store your log records in whatever file format you wish. If you do this, your log browsing procedure must match this file format.

For further information, see the descriptions of the following verbs in the *Network Control Language Reference*:

- &LOGREAD
- &LOGCONT
- &LOGDEL

### Implementing Your Own Logging and Browsing Procedures

If you write you own browsing procedure or your own logging and browsing procedures, you need to implement them.

To implement your procedures, update the LOGFILES parameter group in ICS with your parameter names and then action the group.

## About the Hardcopy Activity Log

A region can have more than one hardcopy activity log, of which only one is open for logging.

### The Activity Log in OS/VS Systems

In OS/VS systems Management Services can be configured to perform logging to disk, tape, or hard copy. From one to nine logs can be specified by including the required number of DD statements in the Management Services execution JCL. Logging can be specified to wrap when the last log is full or is swapped. To obtain the status of these logs, use the SHOW LOGS command.

Input/Output operations are minimized in OS/VS systems through the use of deferred I/O techniques with guaranteed flushing of unwritten log records in the event of system failure.

### The Activity Log in VM/GCS Systems

In VM/GCS systems, the log is directed either to the spooled print file, or to a CMS file.

### Sending Log Information to the Spooled Printer

To send the log to the spooled printer, include the following GCS command in the GCS startup procedure for Management Services:

```
FILEDEF LOG1 PRINTER
```

If you direct the log to the printer, the LOGSWAP command causes the log to be closed and reopened. Each logical division of the log that this creates is identified by an ascending number (LOG0001, LOG0002, ... and so on).

### Sending Log Information to a CMS File

To send the log to a CMS file, include the following GCS commands in the GCS start-up procedure:

```
FILEDEF LOG1 DISK filename filetype filemode
FILEDEF LOG2 DISK filename filetype filemode
...
FILEDEF LOG9 DISK filename filetype filemode
```

where each FILEDEF points to a separate CMS file. The LOGSWAP command then switches the log between the successive log datasets depending upon the number you have defined.

To obtain the status of the logs, use the SHOW LOGS command.

## Format of Logged Information

Each entry recorded on the log is in the following format:

```
12.04.23.12   SMITH     TERM54    +V NET,ACT,ID=NCP001
```

This entry consists of the following information:

- A time stamp in the format HH.MM.SS.TH
- The user ID that entered the command or caused the message to be logged
- The terminal from which the command was entered or to which a message is being sent
- The text of the message or command

Commands are highlighted with a plus sign (+) prefixed to the text to make it easier to distinguish commands from messages when browsing the log. If the command entered is an unsolicited VTAM command, it is highlighted and prefixed with an equals sign (=).

### Format of Logged Timer Initiated Commands

Commands that are executed as the result of a timer initiated command are prefixed by a plus sign, followed by the identity number of the timer command responsible. This is in the format *#nnnn*. For example:

```
15.00.00.01   NETOPER   CNTL01    +#0005 D BFRUSE
```

### Format of Logged Commands Executed in Background Environments

Commands executed under the control of background environments are identified by the following keywords in the user ID field for both the command text and any resulting messages:

| | |
|---|---|
| BG-SYS | Background System Processor |
| BG-MON | Background Monitor |
| BG-LOG | Background Logger |

### Format of Logged Commands from NCL Procedure Dependent Environment

Where a command is executed from an NCL procedure dependent environment (&INTCMD), the node field on the log contains the NCL ID of the process issuing the command.

## Format of the Hardcopy Log

The hardcopy log dataset has the following format:

- A heading on each page—contains the day and date on which the log was created and the system identifier (NMID) of the originating system
- A log identifier on the right side of the page
- 60 lines on each page—this can be altered to suit your requirements using the SYSPARMS LOGPAGE operand. See Appendix D, *SYSPARMS Operands* for details about LOGPAGE.

In OS/VS systems, the log identifier is the DD name under which the log was created. In VM/GCS systems, this is the logical log number (LOG0001, LOG0002, and so on). This log identifier assists log collation after printing.

## Swapping the Hardcopy Log

Swapping the current log allows the log to be freed for immediate printing. To swap the log, use the LOGSWAP command. Swapping the log is only possible when another unused log remains to which logging can continue. You can specify up to nine logs. Logs do not need to be consecutive.

When a log is swapped, the log status, the user ID requesting, and the reason for the swap is recorded. You can display these details with the SHOW LOGS command.

### Swapping the Hardcopy Log in OS/VS Systems

In OS/VS systems each of the logs is identified in the system execution JCL by the LOG DD name followed by a single digit in the range one to nine. For example:

```
//LOG4     DD    SYSOUT=A,FREE=CLOSE
```

Mixing of device types is also valid. Inclusion of FREE=CLOSE allows the log to be printed as soon as it is released by the LOGSWAP command.

### Swapping the Hardcopy Log in VM/GCS Systems

No limit applies to the number of LOGSWAP operations on a VM/GCS system if the log is directed to the printer. However, if the log is directed to CMS files, the limit is nine; each one generates a new logical log identifier in the format:

LOG*nnnn*

where *nnnn* is an ascending number, starting at 0001 and incrementing by one for each LOGSWAP operation.

If the LOG1 FILEDEF statement directs the log to the spooled printer and no other LOG*n* FILEDEF statements exist, then the current section of the log is made available for printing. If the log is directed to a CMS file, LOGSWAP switches to the next FILEDEF statement in the sequence LOG1, LOG2, … LOG9 depending on the number of log datasets defined.

## Wrapping the Hardcopy Log Datasets (OS/VS)

Wrapping allows you to reuse a LOG dataset once all of the available LOG datasets have been used.

The LOGWRAP SYSPARM determines whether log dataset wrapping is allowed. You set the value of this SYSPARM in the Are Activity Logs to Wrap? field when you customize the LOGFILES parameter group in ICS

If you specify NO (the default) in the Are Activity Logs to Wrap? field, then wrapping is not permitted. When all the LOG datasets have been used due to successive LOGSWAP commands, the previous LOG datasets cannot be reused. After the last LOG dataset is used, any further LOGSWAP commands are rejected.

If you specify YES in the Are Activity Logs to Wrap? field, log wrapping is allowed according to the following rules:

- If you are directing your LOG datasets to SYSOUT, then, as each LOG*n* DD card is used, the dataset is dynamically unallocated as a result of the FREE=CLOSE option. In this case, you can reissue an ALLOC command to reallocate another SYSOUT file under the same DD name. For example:

  ```
  ALLOC DD=LOG3 SYSOUT=A FREE=CLOSE
  ```

  This DD name is now available for use as another LOG dataset. Subsequent LOGSWAP operations can now reuse this LOG dataset rather than rejecting the command when the last LOG dataset is used.

- If YES is specified but the LOG DD cards point to sequential datasets, log wrapping overwrites the earlier LOG data held in these datasets. You should take precautions to archive the existing data before allowing the wrap to occur.

## Cross Referencing Hardcopy Logs

To make it easier for operations staff to piece the full log together, certain information is recorded on the last and first lines of LOG datasets that have been swapped.

The first line of a new log that is used in place of a swapped log contains the reason for the swap, or the initiating user ID.

The last message printed on a swapped log is the DD name of the new log (OS/VS) or its logical identifier (VM/GCS). Also printed at the start of the new log is the DD name or logical ID for the previous log.

## Dealing with I/O Errors on the Hardcopy Log (OS/VS)

If an I/O error occurs on a log, the log is closed and the next available log is automatically swapped to, if one is available, and logging continues. This also applies to dataset full conditions when logging to disk.

If the I/O error occurs on the last available log, a warning message is sent to all monitor terminals informing them that logging has ceased and the system is continuing without logging. The STATUS command also includes a warning message if logging has been halted. All log messages continue to be passed to LOGPROC for analysis even if no log output is possible.

## Recording Additional Information on the Hardcopy Log

You can use the LOG command to record additional information on the log that is outside the scope of that recorded by default. For example:

```
LOG =====> REPORT TERMINAL LU345 FOR SERVICE, FAULTY KEYBOARD
```

The message recorded can be up to 256 characters long. Additional messages can also be written to the log using the &WRITE LOG=YES NCL statement from NCL procedures.

## Writing to the System Log (OS/VS)

The SYSPARMS SYSLOG operand can be used to direct Management Services to write all logged output to the system log as well as to its own log, or to write all VTAM PPO messages received to the system log. See the description of the SYSPARMS SYSLOG operand in Appendix D, *SYSPARMS Operands*.

# 13

# Implementing the SOLVE Subsystem Interface (SOLVE SSI)

This chapter provides information about the SOLVE Subsystem Interface (SOLVE SSI) and how to implement it. It also provides information about how to implement the Program-to-Program Interface (PPI) in a SOLVE SSI environment.

**This chapter contains the following topics:**

- About SOLVE SSI
- Implementing SOLVE SSI
- Activating SOLVE SSI
- Terminating SOLVE SSI
- Executing SOLVE SSI Commands
- Using PPI with SOLVE SSI

## About SOLVE SSI

SOLVE SSI is an implementation of IBM's Subsystem Interface (SSI). It provides a general-purpose internal facility which allows programs to communicate with subsystems.

The IBM SSI allows you to perform the following:

- Define function codes to allow subsystems to be informed of system events. For example, end of task or address space, operator command, WTO, and so on.
- Anchor public control block structures so that they can be easily found by any interested party, without requiring any system alterations.
- Remember information across job restarts, for the life of an IPL.

SOLVE SSI supports the Program-to-Program Interface (PPI), which is described in the section, *Using PPI with SOLVE SSI*, on page 13-12. It also supports the use of non-SNA terminals and inter-SSI connections.

## Implementing SOLVE SSI

To implement SOLVE SSI, perform the following steps:

Step 1. Define a 1- to 4-character subsystem ID (SSID) name.

The name must be defined to OS/390 or z/OS in the IEFSSN*xx* initialization member in SYS1.PARMLIB.

**Note**

You do not have to define an SSID as it is defined automatically by SOLVE SSI. See the description of the ADDSSI JCL parameter on page 13-5 for more details.

Step 2. Set up the SOLVE SSI started task.

To do this you must check the JCL, dataset allocations, and initialization parameters generated by the installation and setup process. See the following section, *Checking the JCL, Datasets, and Parameters*, for information on how to do this.

Step 3. Specify the SSID in the SYSPARMS SSID operand or JCL parameter of each Management Services region you want to connect to the SOLVE SSI region.

**Note**

It is recommended that the SSI be started as part of the IPL process. If the PROC name (for a started task) is equal to an SSID in IEFSSN*xx*, then it can be started under the MSTR subsystem (in which case SYSOUT cannot be used).

## Checking the JCL, Datasets, and Parameters

Your product's installation and setup process generates the following:

- JCL to set up the SOLVE SSI started task—in the member *dsnpref*.*ssiname*.JCL(SOLVESSI)
- Dataset allocations—in the member *dsnpref*.SI*vvv*.SSIPARM(SSISYSIN)
- Initialization parameters (see the section, *Checking Initialization Parameters*, on page 13-5)—in the member *dsnpref*.SI*vvv*.SSIPARM(SSIPARMS)

For further information about this process, see the *Unicenter Mainframe Installation and Setup Instructions* .

### Checking the JCL

To check the JCL generated for your product's installation and setup process, review the dataset member *dsnpref*.*ssiname*.JCL(SOLVESSI).

The STEPLIB DD statement must point to a library containing the SOLVE SSI load modules. The library is normally the load library that Management Services also executes from. This library must be APF authorized.

### Checking Dataset Allocations

To check the datasets allocated by the SOLVE SSI started task, review the DD statements in the dataset member *dsnpref*.SI*vvv*.SSIPARM(SSISYSIN).

**SSIIN**

This DD statement must point to a dataset, *dsnpref*.SI*vvv*.SSIPARM(SSIPARMS), that has the following attributes:

- Sequential
- Fixed blocked (FB)
- LRECL=80

Initialization parameters are read from this dataset. It is normally a PDS member; for example, SYS1.PARMLIB(NMSSIPRM). If there are no requirements for this file, a DUMMY dataset can be specified. Any parameters specified in this file can be overridden by the JCL parameter.

When implementing PPI with SOLVE SSI, there is additional JCL that must be specified. See the section, *Activating PPI*, on page 13-14, for more information.

**SSILOG**

This DD statement must point to a dataset, or to a SYSOUT specification. It must have the following attributes:

- Variable
- Blocked
- LRECL=137
- ASA control characters (VBA)

If no blocksize if specified, 6233 is used as the default. If this file fails to open, SOLVE SSI terminates abnormally.

**SSIDUMP**

This DD statement is optional, but recommended. It must point to a SYSOUT specification or a dataset with the following attributes:

- Fixed
- Blocked
- LRECL=133
- BLKSIZE=1330
- ASA control characters (FBA)

A formatted dump is written to this dataset in the event of an SSI ABEND. This dump can be used for problem determination.

**SYSUDUMP**

This DD statement specifies a standard SYSUDUMP that is used to obtain a system dump in the event of an abnormal SSI termination. A SYSABEND or SYSMDUMP DD statement can be included in place of SYSUDUMP.

**Note**

If the JCL parameter SDUMP has been set to YES, then it is recommended that you remove any SYSUDUMP, SYSMDUMP, or SYSABEND statements from the JCL for the SOLVE SSI started task.

## Checking Initialization Parameters

SOLVE SSI has various initialization parameters. These control important execution options and facilities.

The initialization parameters are specified in the SSIPARMS member created during region setup and placed in *dsnpref*.SI*vvv*.SSIPARM(SSIPARMS).

To check the initialization parameters generated, and to add further initialization parameters as required, open the SSIPARMS member of the *dsnpref*.SI*vvv*.SSIPARM dataset.

**Note**

Parameters can be specified more than once. The last specification is used.

### *Initialization Parameter Descriptions*

The following initialization parameters are recognized. Some (identified by a comment in the following list) are automatically included in the SSIPARMS member. The others in the list can be added to the SSIPARMS member.

**Note**

The PPI facility provides several additional parameters. See the section, *Activating PPI*, on page 13-14, for more information.

**ABENDCODE={ <u>4095</u> | *n* }**

Specifies an alternate user ABEND code to be assigned for SOLVE SSI internal ABENDs. The default is 4095. Valid values of *n* are 1 to 4095.

**ABENDMSG={ <u>NO</u> | YES }**

Specifies whether or not various SSI service ESTAE exits write WTO messages if an ABEND is detected. This parameter can be used to track down external interface errors, for example PPI RC 90.

**ADDSSID={ <u>YES</u> | NO }**

Specifies whether the SSID is defined automatically if an SSID is not found when SOLVE SSI starts. If YES is specified, the SSID is defined automatically. If NO is specified, the SSID must be defined manually in the IEFSSN*xx* SYS1.PARMLIB member.

**ARM={ <u>NO</u>| YES }**

Specifies whether or not the SOLVE SSI region registers with the Automatic Restart Manager (ARM). If NO is specified, the region will not register with ARM. If YES is specified, the SOLVE SSI region attempts to register with ARM. If the registration fails, an error message is written to the SSILOG but the region still starts.

**ARMNAME=*elementname***

Overrides the default ARM element name. The default is SVS_*sysnamessid* where *sysname* is the system name and *ssid* is the SOLVE SSI region. ARM requires a unique element name for each SYSPLEX.

**BLIP={ NO | *n* }**

Specifies the wakeup interval, which will prevent S322 time-outs of the subsystem job. Valid values of *n* can be 1 to 60 minutes. This can be important in a testing environment if SOLVE SSI is running as a batch job. If NO is specified, no wakeup timer is set.

**DAE={ NO | YES }**

Specifies whether NMSSI sets the DAE indicator to ON when requesting an SDUMP. This enables these dumps for DAE processing.

**Note**

Current releases of OS/390 and z/OS perform DAE processing regardless of the setting of this parameter.

**DDSUBSYS={ NO | YES }**

Specifies whether or not SOLVE SSI provides services needed by the DD SUBSYS facility.

You need to specify YES if you are using the DD SUBSYS facility.

For further information, see the *SSI DD SUBSYS Support* appendix of the *Automation Services Administrator Guide*.

**ENF={ NO | YES }**

Controls the setup of the MVS ENF listener during SSI initialization. The default, ENF=NO, means that no listener is inserted. If YES is specified, then an ENF listener is inserted and the ENF interface is activated.

**ENFADD=*n***

Activates the nominated ENF code. *n* must be in the range 1 to 255. If *n* does not correspond to a recognized ENF code that NMSSI handles, then a warning message is produced during startup. This parameter can be repeated as necessary.

The valid ENF codes and their initial status are documented in IBM's *OS/390 Authorized Assembler Programming Guide*.

**ENFDEL=*n***

Deactivates the nominated ENF code. *n* must be in the range 1 to 255. If *n* does not correspond to a recognised ENF code that NMSSI handles, then a warning message is produced during startup. This parameter can be repeated as necessary.

**ENFARMWTO={ NO | YES }**

Can be used to activate the old WTO messages that earlier versions of the ENF interface issued for ARM events. The default is NO.

**PROMPT={ YES | NO }**

Specifies whether or not a message is written (WTO) to the system operator if the supplied SSID is not found or is unusable. If YES is specified, the operator has a chance to specify a new SSID, request a retry, or END the SSI. If NO is specified, and the SSID is not usable, the SSI is immediately terminated.

**SDUMP={ NO | YES }**

Specifies whether NMSSI writes a dump to a SYS1.DUMP dataset when an ABEND occurs.

The default is NO, and the dump is written in accordance with the JCL (for example, SYSUDUMP or SYSMDUMP).

Specifying YES suppresses any dump specifications (such as SYSUDUMP) in the JCL (except as noted below), and forces a dump to be written to a SYS1.DUMP dataset. The dump includes symptom strings that aid analysis.

**Note**

The formatted dump (SSIDUMP) is still written.

In some cases, a system dump (for example, a CANCEL DUMP operator command) is written to a JCL-specified location (such as SYSUDUMP) before NMSSI has a chance to suppress it. Therefore, it is recommended that the SYSUDUMP, SYSMDUMP, and SYSABEND statements be removed from the NMSSI JCL.

**SOLVEJOB={ * | *jobname* }**

Specifies the job names of the regions that can connect to SOLVE SSI. If * is specified, no restriction is placed on the regions that can connect to SOLVE SSI. *jobname* specifies the name of a region running under that job name that can connect to SOLVE SSI (it must be a valid MVS job name). This parameter can be specified up to 16 times to nominate the regions that can connect to SOLVE SSI at once.

**SSID={ * | *name* }**

Specifies the subsystem ID (SSID) that this invocation of the SSI is to use.

> **Note**
>
> The SSID does not need to be defined explicitly, but can be defined automatically. See the ADDSSID JCL parameter above for more information.

If an asterisk (*) is specified (the default), the first 4 characters of the SSI jobname are used. If *name* is used, it must be a valid 1- to 4-character name. The first character must be alphabetic or national and the remainder alphanumeric or national.

**STATS={ NO | *n* }**

Specifies the interval at which a SHOW SSISTATS command is issued. Valid values of *n* can be 1 to 60 minutes. The output of this command is routed to the SOLVE SSI log only. This can be useful for following trends in SOLVE SSI usage. If NO is specified, no STATS timer is set.

**UNIX={ NO | YES } (**included in SSIPARMS**)**

Specifies whether to enable the UNIX System Services (USS) shell interface.

If you set this parameter to YES, and your operating system cannot access USS for some reason, then you receive a warning message, but the task starts successfully.

If you have multiple SOLVE SSIs active on a system, then set up the USS shell interface in one SOLVE SSI only. For the other SOLVE SSIs, specify NO or remove this parameter.

**XCF={ NO | YES }**

Enables the SOLVE SSI region to communicate with other SSI regions by using XCF. If YES is specified, the SOLVE SSI region attempts to register with XCF using the group name and member name (see below).

**XCFGROUP=*groupname***

Overrides the default XCF group name that SOLVE SSI uses when registering with XCF. The default is ZSOLVE01.

**XCFMEMBER=*membername***

Overrides the default XCF member name that SOLVE SSI uses when registering with XCF. The default is *sysname ssid*.

**XEVNT={ NO | YES }**

Specifies whether to include Simple One Shot Event Sender support. Only one SSI started task per MVS region can have YES specified.

**XMS={ * | NO | YES }**

Specifies whether or not SOLVE SSI is to set up and use a cross-memory environment for external interfaces (for example, PPI). The following operands can be specified:

- * (the default)—SOLVE SSI uses XMS if it is supported by the operating system. This means that OS/390 and z/OS use XMS by default, and MSP and MSP/EX do not.
- NO—blocking of XMS is allowed in all environments
- YES—XMS is used, but an error occurs if the environment does not support XMS

## Activating SOLVE SSI

SOLVE SSI is activated by setting up a started task or a job, and then submitting or starting it. However, it is recommended that you start SSI as a started task.

If the SSID SYSPARM or SSID JCL parameter is set when the region initializes, the region initiates a connection to the nominated SSI. When this connection is complete, authorized region users can issue SOLVE SSI commands.

### Connecting to SOLVE SSI

For a region to connect to SOLVE SSI, it must first know the SSID to connect to. You do this by specifying the SSID JCL parameter or by using the SSID SYSPARMS commands.

The region can establish an early connection to SOLVE SSI during initialization by the use of the SSID JCL parameter. The syntax of this parameter is:

```
SSID={ NO | * | name }
```

The operands have the following meanings:

**NO**

No connection to SOLVE SSI is attempted. The connection is only started (or attempted) after a SYSPARMS SSID command is issued. This is the default.

*****

Starts or attempts a connection to an SSID with the first four characters of the region's job name.

***name***

Starts or attempts a connection to the specified SSID.

If asterisk (*) or *name* is specified, an attempt to connect to the SSI is immediately made. If it fails, it retries every *n* seconds, depending on the default value of the SYSPARMS SSIRTINT operand.

To allow connection to SOLVE SSI to be attempted, or stopped, at any time, the SYSPARMS SSID operand can be issued by an authorized user. This SYSPARM has the following syntax:

```
SYSPARMS SSID={ NO | * | name }
```

The operands have the following meanings:

**NO**
: Terminates any current or pending SSI connection. If there is no current or pending SSI connection, it does nothing.

*****
: Starts or attempts a connection to an SSID with the first four characters of the region's job name. If a connection currently exists, an error occurs. If a pending connection exists, it is immediately retried with the new SSID.

***name***
: Starts or attempts a connection to the specified SSID. If a connection currently exists, an error occurs. If a pending connection exists, it is immediately retried with the new SSID.

### Specifying the Retry Interval

The retry interval for SSI connection attempts can be altered using the SYSPARMS SSIRTINT command. This SYSPARM has the following syntax:

```
SYSPARMS SSIRTINT={ 30 | n }
```

The default retry interval is 30 seconds. Valid values for *n* are 10 to 600. Each retry writes a monitor message displaying the failure reason.

## Terminating SOLVE SSI

You can terminate SOLVE SSI in any of the following ways:

- By using the SSI STOP command, as described in the next section, *Executing SOLVE SSI Commands*
- By using the operating system MODIFY (F) command, in the format:

  ```
  F SOLVESSI,FSTOP
  ```

- With an ABEND by the use of the SSI $ABEND$ command. This command should be used only when debugging. SOLVE SSI can also be cancelled, although this is not recommended.

**Note**

If you are using cross-memory services, the address space running SOLVE SSI is terminated and is not available until after the next IPL. (This is an IBM restriction.)

## Executing SOLVE SSI Commands

SOLVE SSI provides several commands that can be used for control and to display statistics. These commands can be issued from the following locations:

- From any system console. Command responses are delivered to the issuing console.
- Any suitably authorized Management Services user ID on the connected region. Command responses are delivered to the issuer.
- Internally. The STATS=*n* JCL parameter causes a command to be internally issued in the SSI itself. The output of this command is routed to the SSI log only.

### SSI Command Descriptions

The following commands are provided:

**SHOW SSIEPS**

Displays information about EPS connections and endpoints as seen from this SSI region. The following operands are supported:

| | |
|---|---|
| DETAIL | Displays additional information about each endpoint |
| LINKS | Displays a list of direct links to other endpoints |
| TOPOLOGY | Displays a topology of all links between all endpoints |
| NOTIFY | Displays a list of notifier endpoints for each link in the topology display |
| STATS | Displays additional statistics lines |

**SHOW SSISTATS**

Displays statistics about internal SSI pools. This command is useful for tuning and debugging.

**SHOW SSIUSERS**

Displays a list of all signed-on SSI environments. The list shows the terminal (or console), user ID, and type.

**SSI SIGNOFF**

Terminates the current command/message environment. This is useful for preventing further receipt of unsolicited SSI messages after issuing some SSI commands.

**SSI STATUS**

Displays the status of the SSI. The display includes the version number and PUT level.

**SSI STOP**

Stops the SSI. This command causes the job to terminate. This command cannot be issued during initialization. A default command authority of 4 is assigned to this command.

These commands are basic SOLVE SSI commands only. PPI adds some additional commands. These are described in the section, *PPI Command Descriptions*, on page 13-15.

## Using PPI with SOLVE SSI

PPI provides a general-purpose facility for programs, written in any language, to exchange data. It also provides a facility for any program to forward a generic alert to NetView or a region. The region implementation closely follows the IBM implementation. Therefore, programs written to run with the IBM implementation will work with the region implementation with no changes.

The region implementation of PPI is supported by SOLVE SSI. No special authorization is required to use the PPI, and it does not depend on having NetView or a region running.

### Implementing PPI

PPI is implemented by SOLVE SSI. This allows it to run regardless of the status of the region. This is important, because applications might need to queue data across the PPI even if the region itself is not active.

The region can use PPI regardless of whether it is being provided by SOLVE SSI or NetView. The region need not have a connected SOLVE SSI and the connected SOLVE SSI need not be the PPI owner to use the facilities of PPI.

**Note**

Only the region connected to the PPI-owning SOLVE SSI can register receiver IDs that start with NETV or NETM.

When the region is initialized, it issues a conditional load for CNMNETV. If CNMNETV is found, then PPI facilities are available. This conditional load is issued only on MVS, MSP, and VOS3 systems.

## PPI-specific Parameter Descriptions

Before PPI can be activated, you need to specify PPI-specific parameters in the SOLVE SSI SSIPARMS member in the *dsnpref.rname*.TESTEXEC dataset. The following PPI-specific parameters are recognized. Some (identified by a comment in the following list) are automatically included in the SSIPARMS member. The others in the list can be added to the SSIPARMS member.

**PPI={ YES | NO } (**included in SSIPARMS**)**
Indicates whether this SOLVE SSI is to provide PPI services (YES) or not (NO). Remember that only one SSI can provide the interface.

**PPIFREELIM={ 25 | *nnnn* } (**included in SSIPARMS**)**
Specifies the maximum number of pages (each 4K) of storage that are retained in the PPI data buffer free storage pool. The default is 25 pages (100K). The value can be from 10 to 1,000 pages. Buffers longer than approximately 4,060 bytes are not allocated out of this pool.

The buffer free storage pool is initially empty. Storage is obtained from the system as required, and, as data buffers are received, their storage is returned to the pool. If the number of free pages in the pool then exceeds this limit, the excess pages are freed to the system. The pool reduces overheads by eliminating most GETMAIN/FREEMAIN activity.

**PPIINATO={ *n* | NO | 5 }**
Specifies the number of minutes that must elapse before an inactive PPI user, with nothing queued to it, is not displayed by the SHOW PPIUSERS command without the INACT operand. Valid values are 0 to 1,440, or NO. Zero or NO indicate no time-out. The default is 5.

**PPIMAXQB={ 100000 | *nnnnn* } (**included in SSIPARMS**)**
Specifies the largest value allowed for the PPI receiver queue limit. Values larger than this are rejected. Valid values are 1000 to 1000000. The lowest limit that a PPI receiver can have is 0.

**PPIMAXBL={ 65536 | *nnnnn* }**

Specifies the largest data buffer size that can be queued to a PPI receiver. This allows the setting of a reasonable limit. Valid values are 1,000 to 1,000,000.

**PPINETMR={ YES | NO }**
**PPINETVR={ YES | NO }**

Specifies whether or not PPI receiver names starting with NETM or NETV are to be restricted to being defined by the connected Management Services main task only. The default for both is YES, meaning that the associated name is restricted.

**PPINPREF={ * | *xxxx* }**

Specifies the PPI name service prefix value. An asterisk (*) causes the SOLVE SSI SSID to be used. Otherwise, this must be a 1- to 4-character name consisting of alpha, numeric, or national characters only. It is recommended that the value be the same as the SOLVE SSI-connected Management Services domain ID as specified in the NMDID JCL parameter.

**PPIRC90T={ YES | NO }**

Controls the writing (WTO) of additional debugging messages from PPI whenever a PPI return code 90 is returned on a PPI API call.

More information on return code 90 can be found in the *Management Services User's Guide*.

**PPIREUSE=*n***

Specifies the number of PPIINATO intervals that must elapse before an inactive PPI receiver with nothing queued, that was a PPI-supplied name (function 60), is purged and the name made available for reassignment.

## Activating PPI

PPI is activated via the JCL parameter, PPI=YES | NO, which is specified within SOLVE SSI. If PPI is set to YES, then SSI attempts to activate PPI when SOLVE SSI initializes.

If PPI is not activated, then check:

- Whether another SOLVE SSI owns the PPI. Only one SOLVE SSI can provide PPI services at a time.
- That SOLVE SSI can build the required control block structure to support PPI.

## PPI Command Descriptions

The following commands are available or have extended function when PPI is active:

**SHOW PPIUSERS [ =*name* | =*prefix** ] [ INACT ]**

Displays a list, in receiver ID order, of all defined PPI receivers. The display includes statistics on buffer counts and storage. If active, the owning job name and ASID are displayed.

**SHOW SSISTATS**

If PPI is active, additional statistics on PPI are displayed. This includes the number of receivers and statistics on the PPI buffer storage pool.

# 14

# Recording CPU Usage

This chapter describes how to use the CPU-time accounting facilities of NCL. It covers the following topics:

**This chapter contains the following topics:**

- About NCL CPU-time Accounting
- Implementing NCL CPU-time Accounting
- Starting and Stopping CPU-time Accounting
- Displaying the Status of NCL CPU-time Accounting

## About NCL CPU-time Accounting

NCL CPU-time accounting allows you to 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 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 command. See the section, *Starting and Stopping CPU-time Accounting*, on page 14-5 for more information about this command.

You can also display the current status of CPU-time accounting. See the section, *Displaying the Status of NCL CPU-time Accounting*, on page 14-7, for more information.

## Implementing NCL CPU-time Accounting

To implement NCL CPU-time accounting, the OPT=01 JCL parameter must be specified. Your system must also be running *authorized* in order 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 shutdown 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.

## Adjusting 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.

For details on how to implement the timing of reporting, see *Starting and Stopping CPU-time Accounting*, on page 14-5.

## SMF Record Format

The SMF record format consists of three sections:

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

The format is shown in Table 14-1.

*Table 14-1. Standard SMF Record Format*

| 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) |
| SOLVE Prefix | | | | |
| 18 | 1 | binary | SOLVE | Sub-category: X'06' for user CPU-time accounting |
| 19 | 1 | null | - | Reserved |
| 20 | 12 | character | SOLVE | NMID of SOLVE host |
| CPU-time Accounting Section | | | | |
| 32 | 8 | character | SOLVE | SOLVE user ID CPU-time value applies to |
| 40 | 1 | binary | SOLVE | Code indicating where CPU-time was used (X'00' is NCL CPU-time |
| 41 | 1 | null | - | Reserved |
| 42 | 8 | stck | SOLVE | CPU-time accumulated SOLVE 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) |

## Starting and Stopping 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.

### The USERACCT Command

The USERACCT command has the following syntax:

```
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)
```

The meaning of each operand is as follows:

**START**

Used to activate NCL CPU-time reporting. This operand controls whether or not 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. Valid values are 128 to 255. It is effective immediately after the command is issued, so the next record generated uses this *smf-record-id* value.

If not specified, the default SOLVE SMF record ID is used. If there is no default SOLVE 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 generation of SMF records. Valid values are 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. This operand is only allowed if NCL CPU-time reporting is inactive at the time the command is entered. If this operand is not specified, the default is 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 a value in CPU seconds, with up to 2 decimal places (for example, 1.25 seconds). It is effective immediately after the command is issued, and is applied on the next report. The default is 0.

**SET**

Used to modify 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 only accepted when CPU-time reporting is inactive.

**STOP**

Used to stop user NCL CPU-time reporting. This operand controls whether or not SMF reports on the statistics are to be 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 only accepted 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.

## Displaying 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:

- Whether NCL CPU-time accounting is active or inactive
- 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 100ths of a second.

# 15

# Implementing User Programs

This chapter describes how to use the SUBSYS facility to implement user programs. It covers the following topics:

**This chapter contains the following topics:**

- About the SUBSYS Facility
- Sending Parameter Lists to the Subsystem
- Controlling Subsystems
- Writing a SUBSYS Program

## About the SUBSYS Facility

The SUBSYS facility allows you to call user programs and have them run in a separate subtask. This allows you to:

- Enhance the &CALL NCL verb by attaching a long-running subtask to handle &CALL requests
- Preserve the state of system resources, such as open files, external database interfaces, and so on
- Reduce the system overheads associated with each call

The SUBSYS facility is supported by the SUBSYS command, and extensions to the &CALL NCL verb.

You should not confuse the subsystems provided by the SUBSYS facility with the MVS Subsystem Interface, or other MVS Subsystems. These are purely internal subsystems managed by Management Services.

### Using &CALL Without SUBSYS

If you are using &CALL without the SUBSYS facility, the following disadvantages are encountered:

- An MVS ATTACH is issued for each call. This means that the called program cannot easily remember things across calls.
- Significant CPU time is expended in ATTACH processing
- Continual opening and closing of files by MVS
- Use of either &LOCK (in NCL) or ENQ/DEQ (in assembler) to serialize

### Using &CALL With SUBSYS

To counter the disadvantages of using &CALL, you can use the SUBSYS facility. This provides the following:

- The target program is only attached once and all work is queued to it
- The target program can easily keep files open
- The target program is informed of NCL process termination, and of subsystem start and stop. It is also notified of system shutdown.

The SUBSYS program is attached once, as a long-running task so that you can open files, get storage, and so on. These resources are not released when you return.

### Extensions to &CALL

The SUBSYS command allows you to nominate a subsystem as a complete &CALL replacement. This means that any existing NCL procedures that have used &CALL to call a program, for example X, can be automatically redirected to a SUBSYS defined as X. This is one way to gain immediate improvements in performance for some programs.

The &CALL verb is fully described in the *Network Control Language Reference*.

### Uses of SUBSYS

Some examples of the use of the SUBSYS facility are:

- Access to foreign DBMS
- The ability to read and write non-supported file formats from NCL
- High-performance reworks of existing &CALL programs
- Interfaces to some IBM utilities such as IDCAMS. One example is UTIL0035.

## Sending Parameter Lists to the Subsystem

When a call is made to an NCL procedure using the SUBSYS facility, parameters are passed to that procedure. These parameters are passed in a list which can be in an old or a new format. The old parameter format is described in the *Network Control Language Reference* in the &CALL description.

The new parameter list format is available for direct call-attach and is specified in the PARMLIST=NEW operand of the SUBSYS DEFINE command. The new format provides the following:

- Called programs can determine whether or not they have been called with an old or new format list
- Provides more information about the environment. There are sections that describe the current Management Services, USER, and NCL process.
- Additional NCL process termination, initialization, and shutdown calls
- NCL process correlators which allow you to be independent of the shared NCL process correlator by writing reentrant SUBSYS code

The PARMLIST format is described in the distributed macro, $NMNCPL.

**Note**

The new format PARMLIST is compatible with high-level languages that use standard IBM linkage conventions.

## Controlling Subsystems

In order to control your subsystems you need to be able to perform various actions on them. Each subsystem can be:

- Defined
- Started
- Stopped
- Deleted
- Reloaded
- Have its status displayed

### Defining a Subsystem

To define a subsystem to Management Services, use the SUBSYS DEFINE command. When this command is issued, the nominated subsystem is defined and the indicated program is loaded into storage. By default, the subsystem is also started. To prevent a subsystem being started when being defined, specify the NOSTART parameter.

For detailed information about how to use the SUBSYS DEFINE command, see the *Management Services Command Reference*.

### Starting a Subsystem

A subsystem must be started before it can accept work. To start a subsystem, use the SUBSYS START command. This command is used to start a defined or inactive subsystem. A subsystem can also be automatically started when it is defined.

When a subsystem is started, the following processing occurs:

1. A small part of Management Services is attached
2. An initialization of the attached Management Services code occurs
3. If the subsystem has PARMLIST=NEW in effect, an initialization call is made to the subsystem code [NCPFFUNC = NCPFFSIN (8)].

The subsystem is now allowed to accept calls (&CALL statements).

## Stopping a Subsystem

A subsystem can be stopped. This allows it to stay defined, but calls are no longer permitted. To stop a subsystem, use the SUBSYS STOP command. When the SUBSYS STOP command is entered, the following occurs:

1. If the subsystem is defined with PARMLIST=NEW, a stop call [NCPFFUNC = NCPFFSTM (12)] is made to the program. This allows the program to clean up any important resources.
2. The subsystem is detached
3. Any pending &CALL requests are rejected. No further &CALL requests are permitted.

A stopped subsystem can be restarted by the SUBSYS START command.

### Forcing a Subsystem to Stop

Sometimes a subsystem hangs in the user code. This could be caused by a loop, or an unsatisfied WAIT. Regardless of the reason, all NCL processes waiting on a call may wait indefinitely. This is especially likely while developing subsystem code. To find out whether the subsystem has hung, use the SHOW SUBSYS command.

You can stop a hung subsystem with the SUBSYS FORCE command. The subsystem is force-detached, cleaned up and then assumes the STOP status.

## Deleting a Subsystem

A subsystem definition can be deleted by using the SUBSYS DELETE command. You must stop the subsystem before it can be deleted. Any following requests to this subsystem are rejected.

## Reloading the Program

If, while developing a subsystem, a new version of the program is required, then the SUBSYS RELOAD command can be used, as an alternative to STOP/DELETE/DEFINE.

The indicated subsystem must be stopped. The program is deleted (provided that it has no other users), and a new copy is requested. The subsystem can then be restarted by using the SUBSYS START command.

**Note**

If the program has other users, the delete does not work, and the reload obtains the same version as before.

### Displaying the Status of a Subsystem

To display the status of all defined subsystems, use the SHOW SUBSYS command.

## Writing a SUBSYS Program

When you write a SUBSYS program it needs to be able to handle the following events:

- An NCL process that called the subsystem terminates
- There is an initialization call
- A SUBSYS STOP command is executed
- Management Services enters shutdown
- There is a subsystem clean up

To make it easy to track data associated with a given environment (such as an NCL procedure), various correlator words are provided from which you can anchor control block structures.

The following sections describes how to write a SUBSYS program using the new parameter format list. For information on how to write a program using the old parameter list, see the &CALL description in the *Network Control Language Reference*.

### Subsystem Program Considerations

When writing a subsystem program, the following should be considered:

- Use the new format parameter list. Remember that you can validate the format by checking that the value pointed to by R1 on entry is equal to R1.

  ```
  C    R1,0(,R1)
  BE   NEWFMT
  B    OLDFMT
  ```

  The following facilities are only available if the new parameter list format is used:

  - An NCL process cleanup call
  - System initialization calls
  - System shutdown calls
  - Private correlator calls

- Check whether your program has been called as a subsystem rather than an &CALL attach. If an asterisk is found in the subsystem name field in the (new) parameter list, then it is an &CALL attach, not a SUBSYS call.

- A subsystem gets a private correlator word for each unique NCL process that calls it. This correlator word can be altered (it starts as 0) and is remembered and returned on the next call for that NCL process. It is an ideal place to anchor process-related control blocks. The shared correlator is shared with all subsystems or other &CALLed programs for this NCL process.
- Any NCL process that calls a subsystem using the new parameter list causes a cleanup call to be provided to that subsystem. This call is made regardless of the type of NCL process termination (normal, abnormal, and so on). The private correlator is provided, and this allows the subsystem to clean up any control blocks as required. If an NCL process is flushed while on a SUBSYS call, the subsystem is not notified of the fact until later. It need not worry about the process disappearing while actually processing the current call.
- A SUBSYS DEFINE (without NOSTART) or SUBSYS START command sends an initialization call to the subsystem. This allows the subsystem to initialize its environment. It should anchor any control blocks in the SUBSYS correlator in the new format parameter list.
- A SUBSYS STOP sends a termination call to the subsystem program. This allows the subsystem to clean up any control blocks, and so on, prior to being detached.
- If Management Services is shut down, the subsystem is sent a system shutdown call. It must quickly clean up its environment. It is force-detached several seconds later regardless of whether it is finished or not.
- Only as many parameters as were passed can be returned to a caller. This is the same as the previous &CALL. Each parameter is limited to 256 characters of data. Binary values are preserved both on input and on return. Be sure to set the lengths correctly.
- All calls to a specific subsystem are queued and processed in turn. This means that, if the subsystem code takes a long time to process a call (for example, issuing a WAIT on VTAM input), all pending calls are also delayed. This is a restriction of the current implementation.
- This can be avoided by queuing requests to the subsystem to a manager process (using the INTQ command and a globally known NCLID). This manager process talks to the subsystem on behalf of all users of the subsystem. Then the subsystem code can wait on many events and return the first event to the manager, which can then pass it back to the requestor.
- If the subsystem ABENDs, the subsystem will stop. It can be restarted.
- SUBSYS FORCE is useful when a subsystem hangs.
- The subsystem load module must reside in a library accessible to Management Services. Typically this is the STEPLIB. Remember that Management Services libraries must be authorized. This means that any libraries in this STEPLIB concatenation must also be authorized. This is a security consideration.

- 31-bit mode programs are supported. All supplied control blocks (for example PARMLIST) are always below the 16mb line.
- If performing multiple functions, use the first parameter passed, as a function indicator:

```
&CALL SUBSYS=SUB1 OPEN ....
&CALL SUBSYS=SUB1 READ ....
&CALL SUBSYS=SUB1 READ ....
&CALL SUBSYS=SUB1 CLOSE ...
```

## Writing a Subsystem Program in High-level Languages

The new format parameter list makes it possible to write both one-time programs using the &CALL command and subsystems in high-level languages, for example COBOL and PL/1.

There are several considerations for high-level languages:

- Always use a new format parameter list. The old format is incompatible with high-level languages.
- In COBOL, define the input parameters in the LINKAGE SECTION. COBOL cannot easily handle a variable number of parameters, but as long as the extra parameters on a specific call are not referred to (that is, the count is obeyed) then no problems should arise. Another technique may be used to access a variable number of parameters. The individual parameters are actually in contiguous storage. Thus, instead of individual parameters, just define the first one, as an array:

```
LINKAGE SECTION.
... other parms
01 NCPC.
03 NCPCCNTPIC S9(9) COMP.
01 PARMS OCCURS 1.  ... actual count is NCPCCNT
03 LENPIC S9(9) COMP.
03 VALPIC X(256).

PROCEDURE DIVISION USING PLIST ... ... NCPC PARMS.
```

- Dope vectors are not set up for PL/1. Because of this, define all individual input parameters as FIXED BIN(31) and BASE a structure over them:

```
DCL NCPN_I           FIXED BIN(31),
    1  NCPN          BASED ADDR(NCPN_F),
        3  NCPNPROC  CHAR(8),
```

  The suggestion for COBOL regarding variable parameter counts applies to PL/1 also. Since the actual number of parameters is not part of the pseudo-array, a REFER structure cannot be used. Use PARMS(1) on the DCL.

  Do not compile with SUBSCRIPTRANGE in effect.

- Use the RETURN-CODE variable in COBOL to set the return code. Use CALL PLIRETC (code) IN PL/1.
- Use GOBACK rather than STOP RUN in COBOL to return.
- Both COBOL and PL/1 have large overheads for establishing the run-time environment. When repeated calls are necessary, these overheads can be excessive. Consult the relevant programmer's guide for details on callable interfaces that can perform a one-time build of this environment.

# 16

# Tuning System Performance

This chapter describes how to tune Management Services so that you gain the most effective utilization of the functions and resources available. It covers the following topics:

**This chapter contains the following topics:**


- About Enhancing Performance
- Tuning at the System Level
- Distributing Management Services Functions Across Different Domains
- Tuning Your VTAM Interface
- Tuning Panel Use
- Improving NCL Procedure Usage
- Tuning Your VSAM Datasets
- Tuning Communication Between Domains
- Controlling Message Flow in OCS
- Performance Considerations When Writing NCL

## About Enhancing Performance

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

- System
- Application
- Function

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

## Tuning at the System Level

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

- For OS/390 and z/OS systems, the performance group and dispatching priority can affect the performance of Management Services. High-priority systems should be in the same domain as VTAM, and just below it in dispatching priority.
- If you are making heavy use of EASINET and MAI, Management Services should be given a high priority.
- Systems using SOLVE:Central or NetMaster for SNA are typically end-user applications. These systems must be regarded as equally important as CICS or IMS. All system tuning rules that apply to other on-line systems should be used with these systems (for example, for storage fencing), to ensure that Management Services is not starved for real storage.
- Dataset placement is important to consider when increasing the performance of your system. The PANELS, MODS, and NCL COMMANDS datasets in Management Services can have a large amount of I/O activity on them. For this reason, it is wise to place these datasets on volumes and paths that are not otherwise busy. Ways of reducing the I/O activity on these datasets are described later in this chapter.
- Running Management Services as non-swappable can help to increase response time. This is important, particularly in EASINET systems where the overall activity rate may be low, but where quick response is extremely important. The SYSPARMS NONSWAP=YES operand is used to specify whether Management Services is non-swappable.

  **Note**

  Some features require Management Services to be non-swappable.

## Distributing Management Services Functions Across Different Domains

Distributing Management Services functions across different domains can help to increase the performance of your products. The effectiveness of this, however, can depend on the work load of these functions. The following points provide usage information about functions to help you decide which approach is best for your installation:

- EASINET—consumes little system resource, except at network startup. It requires fast response times and should be given a very high priority. If other Management Services usage is not as important, then EASINET is a good candidate for a separate domain.
- MAI—large systems need quick response times. These systems tend to be very active, so MAI is also a good candidate for a separate domain. This domain can then be assigned a performance group that favours it.
- Infrequently-used functions—can be relegated to a separate system that can be downgraded in performance.
- NCL-based systems—are heavy CPU users. Splitting several NCL systems across several Management Services domains can prevent response time bottlenecks caused by excessive CPU resource queueing.

*Example*

You could split up your systems into the following domains:

- VTAM network solicitor (EASINET)—same domain as VTAM with high priority
- MAI region—reasonably high priority, below VTAM, but above applications
- INFO/MASTER and other Management Services applications—same priority as other application regions (for example, CICS/IMS)
- Testing systems—low priority or testing machine

### Using JCL Parameters to Include and Exclude Functions

If you split your Management Services load across several domains, consider using the PROD operand on the JCL parameters. This operand allows the selective inclusion of features. See Appendix B, *JCL Parameters*, for more information on the PROD operand.

For example, an EASINET system could exclude NTS or other features not relevant to you. This means that the related Management Services load modules for these features are not loaded. This can reduce the virtual storage requirements of these domains and increase the amount of below the line storage.

**Note**

JCL parameters are limited to 100 characters, so some include/exclude options will not fit.

### Extending Your JCL Parameter Capacity

To extend your JCL parameter capacity, use UTIL0028. This utility removes the 100 character limit on parameter data. In addition, UTIL0028 can dynamically allocate system datasets based on a control member (for example, in PARMLIB). See Appendix C, *The UTIL0028 Utility*, for further details.

**Note**

The UTIL0028 utility is only available in OS/VS systems.

## Tuning Your VTAM Interface

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

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

**Warning**

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.

## Tuning Panel Use

In systems that are running NCL-based applications, a large number of panels might need to be displayed. These panels are stored on a VSAM dataset called PANELS. Panels are retrieved from this dataset as required. Panel definitions can be retained in storage for both reuse and automatic sharing.

To reduce activity on the PANELS dataset, you can specify the number of panels to be retained in storage for reuse. This value is set by the SYSPARMS MAXPANL operand. The default is 50 panels.

## Tuning Access to Panels

To discover the number of panels that can be retained in storage without adversely affecting the performance of your system, use the SHOW PANELS command. This command provides statistics on the effectiveness of the Retained Panels List.

By increasing or decreasing the value of the SYSPARMS MAXPANEL operand, and monitoring the SHOW PANELS output to gauge the effectiveness of the increase, you can set a limit that is most appropriate for your system.

**Note**

Only the panel skeleton is retained. This is different from the panel images retained for panels currently being displayed. For example, 300 EASINET terminals displaying the same panel all share the one skeleton, but have 300 different instances of that display.

## Tuning 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 Management Services 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 datastream. The default is 8Kb.
- PANLBUFF—sets the maximum number of pages of storage to be allowed for panel buffers

See Appendix D, *SYSPARMS Operands*, for a detailed description of these SYSPARMS.

To avoid severe response time problems when you have a large number of Management Services 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.

**Note**

An MAI session can cause panel sends from the target system to hang. This is because MAI does not send an acknowledgment to the sending application until the latest screen image has been actually displayed at an MAI user's terminal. This could mean the panel send will not complete for a long time.

You can use the SYSPARMS MAIRESP operand to overcome this problem. See the section titled *Performance Considerations When Writing NCL*, on page 16-16, for more information.

## Monitoring Panel Send Storage Limits

To monitor how your system is performing 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 maybe close to, or above, the defined maximum storage, is that some of the terminals which 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**

The SYSPARMS PANLBUFF command can be issued at any time to increase or decrease the active panel send limit. This allows dynamic alteration of the throttling characteristics. For example, to minimize the effect of panel sends not completing, you could increase the panel send limit after EASINET startup.

## Reducing the Amount of Storage Needed When Using EASINET

To reduce storage requirements for panels in systems with a large number of EASINET terminals, code #OPT LSM=NO in the EASINET logo panel definition. The LSM option inhibits the maintenance of the last panel image sent to the terminal for all these terminals.

In most cases, an EASINET interaction involves the input of data, followed by a terminal being signed on to an application. In this case, the need to resend a full panel display is outweighed by the storage saved. The trade off is that a broadcast needs to send out entire panels, as no memory of the original panel is retained.

**Note**

The LSM operand is not documented in the #OPT description.

## Improving NCL Procedure Usage

With NCL systems such as NetMaster for SNA, SOLVE:Central, or any user-written applications, the main concern is the rate at which NCL procedures are being 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 being issued as NCL executes.

For example, TSO CLIST performance used this way, creates the following problems:

- 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

### Monitoring 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

### Sharing NCL Procedures Between Users

To help reduce the problems of NCL procedures being loaded too frequently, share them between users. NCL code is reentrant, and can be shared by any number of concurrent users.

To specify that procedures can be shared, specify the SYSPARMS NCLPRSHR operand as YES. This operand specifies that any NCL procedure that is in storage can be shared by any number of concurrent users, and is only released from storage when the use count becomes zero.

NCLPRSHR also allows you to set a number of NCL procedures to be shared. A value of 30 to 50 allows a dramatic reduction in NCL procedure loading.

See Appendix D, *SYSPARMS Operands*, for a detailed description of the NCLPRSHR operand.

### Specifying a Retention Storage Limit

You can use the SYSPARMS NCLPRSHR=*nnn*K operand to set a retention limit in storage, rather than a number of NCL procedures. Specifying a storage limit can significantly reduce the number of NCL procedure loads, with a substantial reduction in CPU usage. The default is NCLPRSHR=30 for all systems.

## Preloading NCL Procedures

You can nominate a set of NCL procedures to remain preloaded, using the SYSPARMS PRELOAD operand. This involves a once-only load and precompile. These procedures can then be shared by any number of users. Preloading NCL procedures is useful when you have routines that are short, and the chance of several concurrent users is slim. These procedures are constantly flushed from storage and reloaded again. It is also best to specify any heavily-used NCL procedures to be preloaded.

See Appendix D, *SYSPARMS Operands*, for a detailed description of the PRELOAD operand.

## Other NCL Procedure Considerations for Increasing Performance

Other ways that can help NCL loading performance include:

- Specify a large BLKSIZE for the NCL libraries
- Use short concatenations. Long concatenations slow down the BLDL for the member.

## Tuning Your VSAM Datasets

Many Management Services applications rely on VSAM UDBs and NDBs. The way you manage your VSAM datasets has a direct effect on the performance of your Unicenter, NetSpy, and SOLVE products. The following facilities are provided by Management Services to allow you to tune UDBs and NDBs for maximum performance benefits:

- Buffer sharing
- Deferred write capabilities
- VSAM processing in a subtask
- Record size management
- Database activity management
- Performance monitoring facilities

These facilities are described in the following sections.

### Using Buffer Sharing

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

To allow buffer sharing for a VSAM dataset, use the UDBCTL OPEN command with the LSR operand.

See the *Management Services Command Reference* for detailed information about the UDBCTL command.

A sample LSR pool definition is distributed in $SANMLSR. This sample is invoked by the distributed procedure, NMINIT. This sample serves as a starting point from which to build the settings that best suit your system.

When specifying the LSR pool definition that is best suited 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 command. For details, see the *Management Services Command Reference.*

## Using Deferred Write Capabilities

Deferred write capabilities allow you to defer the updating of datasets 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 dataset to have deferred write capabilities, specify the DEFER operand on the UDBCTL OPEN command.

See the *Management Services Command Reference* for detailed information about the UDBCTL command.

## Performing VSAM I/O in a Subtask

You can nominate VSAM I/O to be performed by a subtask rather than the Management 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.

See Appendix B, *JCL Parameters*, for more information about specifying the VSAMIO JCL parameter.

You can use the SHOW VSAMIO command to obtain statistics of VSAM processing. Figure 16-1 is an example of SHOW VSAMIO command output.

*Figure 16-1. SHOW VSAMIO Command Display*

```
(18.51)---------- NetMaster Operator Console Services (PROD) -----------------
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*
_____________________________________________________________________________




-------------------------------- NetMaster --------------------------------
==> show vsamio
```

## Tuning Record Sizes

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. For details on the storage needed by an NDB record, see the *Network Control Language User's 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 allows data and key record alterations in the NDB structure to take place without causing excessive numbers of CI/CA splits.

As well as these precautions, you should regularly reorganize your databases out and in with REPRO.

**Note**

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

## Tuning Database Activity

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 dataset 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.

**Warning**

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

## Monitoring VSAM Tuning

The SHOW VSAM command is provided to display attributes and statistics about VSAM databases. Figure 16-2 is an example of a SHOW VSAM display.

*Figure 16-2. SHOW VSAM Command Display*

```
(16.26)-------------- Operator Control Services (PROD) ------------------------
 SHOW VSAM
 N15101 DDNAME   RECSZ D-CISZ I-CISZ CI-SP CA-SP D-BF I-BF STRSH BFRSH LSR CTL
 N13522 VFS       4096   4608   2048     3     0    4    5     0     0 NO  DSN
 N13522 USERIDS   4000   4096   1536     0     0    4    3     0     0 NO  DSN
 N13522 OSCNTL   32700  32768   4096    42     2    4    3     0     0 NO  DSN
 N13522 NCLLOG1    320   4096   2048     0     0    0    0     0     0 YES DSN
 N13522 MODSUSR   4096   8192   2048    17     1    0    0     0     0 YES DSN
 N13522 MODSTST   4096   8192   2048  2826    57    0    0     0     0 YES DSN
 N13522 MODSDIS   4096   8192   2048  1425    26    0    0     0     0 YES DSN
 N13522 MODSINT   4096   8192   2048    58     1    0    0     0     0 YES DSN
 N13522 PANLWRK   4089   4096   2048     0     0    3    3     0     0 NO  DSN
 N13522 PANLTST   4089   4096   2048    33     2    3    3     0     0 NO  DSN
 N13522 PANLDIS   8185   8192   2048    88     2    3    3     0     0 NO  DSN
 N13522 MLPANL    4089   4096   2048     0     0    3    3     0     0 NO  DSN
 N13522 SAMPANL   4089   4096   2048     0     0    3    3     0     0 NO  DSN
 N13522 PLAYPANL  4089   4096   2048     0     0    3    3     0     0 NO  DSN
 N13522 STAGPANL  8185   8192   2048     0     0    3    3     0     0 NO  DSN
 N13522 PANLNDE   4089   4096   2048     0     0    3    3     0     0 NO  DSN
 N13522 NETINFO   2048   4096   2048    11     1    0    0     0     0 YES DSN
 N13522 NEWSFIL   4080   4096   2048  4293    38    0    0     0     0 YES DSN
 N13522 TRFILE    4089   4096   2048     0     0    0    0     0     0 YES DSN

 AUTOHOLD ------------------------ NetMaster-----------------------------------
 ==> SHOW VSAM
```

### Monitoring 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.

For example, Figure 16-3 displays LSR pool statistics from a system that has been customized, and as a result has significantly more buffers than the distributed definition.

*Figure 16-3. SHOW LSR Command Display on a Customized System*

```
(18.54)---------- NetMaster Operator Console Services (PROD) ------------------
SHOW LSR
N15A30 ACTIVE LSR POOL 0 STATISTICS
N15A31 KEYLEN: 255 STRNO: 220 FIXIOB: NO FIXBFR: NO STRMAX: 12 ACTIVE: 22
N15A32   SIZE COUNT  P. READS BUF FOUND       UIW      NUIW %FOUND  HS-COUNT
N15A33    512    20         7       823         0         0  99.15
N15A33   2048    80       475    217714      3597         0  99.78
N15A33   4096    80       157    688944       225         0  99.97
N15A33   8192    30     35838    169602    103924      2517  82.55
N15A33  10240    20         0         0         0         0   0.00
N15A33  12288    26     16736    115875      2391         0  87.37
N15A33  16384     8        24        74         0         0  75.51
N15A33  28672     6        40     34858       549         0  99.88
N15A30 ACTIVE LSR POOL 1 STATISTICS
N15A31 KEYLEN: 255 STRNO: 250 FIXIOB: NO FIXBFR: NO STRMAX: 2 ACTIVE: 1
N15A32   SIZE COUNT  P. READS BUF FOUND       UIW      NUIW %FOUND  HS-COUNT
N15A34 DATA...
N15A33   8192   500       273     40077      4871         0  99.32        50
N15A35 INDEX...
N15A33   2048   100         8     79640         0         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         1       237         0         0  99.57
------------------------------- NetMaster ---------------------------------
==> 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 12K and 16K-byte pools found. This is an indication that the number of buffers could be increased for those pool sizes.

## Monitoring Dataset 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. Figure 16-4 is an example of SHOW NDB=ALL command output.

*Figure 16-4. SHOW NDB=ALL Command Display*

```
(16.26)---------- NetMaster Operator Control Services (PROD) ------------------
SHOW NDB=ALL
N89503 NAME     STATUS     QCMD  USERS CMDS-DONE SUBT QSUB DFR LSIZ LHWM TSILJ
N89504 ZOSSEC   ACTIVE        0      0       932    1    0 NO   201  201 NNNNN
N89504 ZAMSFIL  ACTIVE        0      0        88    0    0 NO    40    8 NNNNN
N89504 ZAMTFIL  ACTIVE        0      0        41    0    0 NO    40    9 NNNNN
N89504 ZMGDATA  ACTIVE        0      0      3272    3    0 NO    40   32 NNNNN
N89504 ZMGTEXT  ACTIVE        0      0       131    1    0 NO   168  168 NNNNN
N89504 ZMGHIST  ACTIVE        0      0       913    1    0 NO    40    8 NNNNN
N89505 *END*
_______________________________________________________________________________









 --------------------------------- NetMaster ---------------------------------
 ==> SHOW NDB=ALL
```

## Tuning Communication Between Domains

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

### Tuning 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 allows logmodes to be used with pacing turned on. This can prevent a VTAM link being overrun by INMC traffic.

See Appendix D, *SYSPARMS Operands*, for details about the INMCBFSZ operand. See Chapter 6, *Communicating Between Domains Using INMC*, for information about changing the RUSIZE.

## Tuning 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

## Tuning FTS

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

- The block sizes of the transmitted datasets
- 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 FTS traffic. FTS has the lowest INMC priority. This is not adjustable.

The only internal tuning that can be performed in FTS is to limit the number of initiators for a specific destination to prevent excessive resource commitment. This can be done by the FTS initiator function, or through the FTSINIT command. See the *Management Services Command Reference* for details.

## Tuning 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.

## Controlling 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 via 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. See *Improving NCL Procedure Usage*, on page 16-7, for more information on using 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 SYSPARMS MODLOAD command, if the module is reentrant, to load it once into storage
  - Use the SUBSYS facility to attach the program once

## Using Commands to Control Resource Consumption by NCL

There are several commands that can be used to alter the performance of Management Services. These system performance commands include SYSPGT, SYSRCT, SHOW SYSWAIT, SHOW SYSPGT, and SHOW SYSRCT. See Appendix N, *Performance and Tuning Commands*, for a full description of these commands. 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 Management Services to run with a high priority in the operating system, but prevents it 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. See the *Management Services Command Reference* for a full description of the SHOW NCL command.

**Warning**

Extremely busy system procedures that do not wait because there is always a message available for &*xxx*READ, 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)
```

# 17

# Using the SYSCMD Facility

This chapter describes how to use the Management Services SYSCMD facility in different environments.

**This chapter contains the following topics:**

- Using SYSCMD in an OS/390, z/OS, MSP, or VOS3 Environment
- Using SYSCMD in an &INTCMD Environment
- Prefixing Messages
- Facilitating User-Generated Messages
- SYSCMD Command Summary
- SYSCMD NCL Verb and System Variable Summary

For information about the SYSPARMS command and associated operands used to tailor and control the SYSCMD facility, see Appendix D, *SYSPARMS Operands.*

## Using SYSCMD in an OS/390, z/OS, MSP, or VOS3 Environment

This section describes how you can use the SYSCMD command in an OS/390, z/OS, MSP, or VOS3 environment.

### Setting Console Timeouts

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.

For details of the timeouts that you can use, see the descriptions of the AOMCTO1, AOMCTO2, and AOMCTO3 operands in Appendix D, *SYSPARMS Operands*.

### Locking 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.

There are, however, cases where a specific environment cannot afford to miss out on obtaining a console; a situation that can occur if the pool of consoles is temporarily exhausted or the consoles are all within timeouts and are unable to be reassigned.

You can lock a console, or you can use an NCL procedure to lock a console to the environment.

To lock a console, use the SYSCMD OPT=LOCK command. The console is then owned by you and cannot be taken for use by another user.

If you use the NCL procedure to lock a console, then also set the NCL procedure to issue &INTCLEAR TYPE=ANY to clear pending messages *without* destroying the environment. Otherwise, if the environment is destroyed, the console will be released.

#### Clearing a Lock

Once 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, causing the dependent processing environment to be terminated.
- You use the AOM STOP command to stop SYSCMD processing.

## Simulating Command Issue from a Specific Console

If you want to simulate the issue of 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 within the system.

As 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 OS/390 modify command to send a command to Management Services:

```
F nm,XYZ
```

Console 15 is signed on to Management Services as *xxxx*CN15 (where *xxxx* is the system user prefix). The XYZ command is then executed in this signed-on environment. Since XYZ is not a Management Services command, an NCL procedure of the same name is loaded and executed. If the following statements are in XYZ, then the commands are sent to OS/390 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 also 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.

## Using SYSCMD in an &INTCMD Environment

A very powerful feature of systems is the ability to execute any Management Services command in a *dependent processing environment*.

To do this, use the &INTCMD NCL statement. The results of commands issued in this way 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.

### Clearing 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.

## Prefixing 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.

- MVS job name

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

- MVS job ID, in the format JOB *nnnnn*, or STC *nnnnn*, or TSU *nnnnn*

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

- Management Services message source domain

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

For more information about message attributes, see the relevant operands in Appendix D, *SYSPARMS Operands*.

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

## Facilitating 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 within NetMaster for SNA 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.

### Using the &WTO NCL Statement

The &WTO NCL statement provides a direct interface to the WTO macro or SVC. It can be used by any NCL procedure executing in Management Services.

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 these recommendations:

- 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 for &WTO

Suggested uses of &WTO are:

- When a major problem is detected—make the message Non-Roll-Delete, 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.

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

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

- Issue a console broadcast, for example:

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

## Using the &WTOR NCL Statement

The &WTOR NCL statement provides a direct interface to the WTOR macro or SVC. It can be used by any NCL procedure executing in Management Services.

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

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 these recommendations:

- 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 to a minimum the number of WTOR messages outstanding at any one time.

### Suggested Use for &WTOR

A suggested use for &WTOR is to enable any NCL procedure to carry on a dialog with the MVS console operators:

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

## Using the &DOM NCL Statement

The &DOM NCL statement provides a direct interface to the DOM macro or SVC. It can be used by any NCL procedure executing in Management Services.

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

A DOMID is formatted as eight hexadecimal digits. The first two are the system ID and the last six are an OS/390-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 recommendations:

- 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 for &DOM

A suggested use of &DOM is that you use it 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. For example:

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

## SYSCMD Command Summary

**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 OS/390, z/OS, MSP, or VOS3 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.

For detailed information on the above SYSCMD facility commands, see your *Management Services Command Reference* manual.

## SYSCMD NCL Verb and System Variable Summary

The following are verbs and built-in functions:

**&DOM**
Generates an MVS DOM (Delete-Operator-Message).

**&WTO**
Generates an MVS or WTO (Write-To-Operator) message.

**&WTOR**
Generates an MVS WTOR (Write-To-Operator with Reply) message and waits for a reply.

The following are system variables:

**&ZMAOMAU**
AOM message authorized issuer flag

**&ZMAOMBC**
AOM message broadcast flag

**&ZMAOMDTA**
Indicates whether AOM data is present..

**&ZMAOMID**
AOM ID

**&ZMAOMJI**
AOM job ID for MVS-sourced messages

**&ZMAOMJN**
AOM job name for MVS-sourced messages

**&ZMAOMMID**
AOM message ID

**&ZMAOMMIN**
AOM minor line flag

**&ZMAOMMLC**
YES if AOM minor CTL line

**&ZMAOMMLD**
YES if AOM DATA

**&ZMAOMMLE**
YES if AOM END

**&ZMAOMMLL**
YES if AOM LABEL

**&ZMAOMMLT**
Has AOM minor line type

**&ZMAOMMLV**
AOM message level

**&ZMAOMMSG**
Indicates if message was AOM generic delivered.

**&ZMAOMRC**
AOM routing codes in list format

**&ZMAOMRCM**
AOM routing codes in MASKCHK format

**&ZMAOMRCX**
AOM routing codes in hexadecimal format

**&ZMAOMSOS**
AOM message source operating system environment

**&ZMAOMTM**
AOM message time

**&ZMAOMTYP**
AOM message type

**&ZMAOMUFM**
AOM user flags in MASKCHK format

**&ZDOMID**
Management Services or MVS DOMID

**&ZFDBK**
Feedback code set by several NCL verbs

**&RETCODE**
Return code set by several NCL verbs

For detailed information on the above SYSCMD NCL verbs and system variables, see your *Network Control Language Reference* manual.

# 18

# Implementing the Data Space Manager

This chapter provides information about the Data Space Manager and how to implement it.

**This chapter contains the following topics:**

- About the Data Space
- About the DIV Dataset
- About the Data Space Manager
- Implementing the Data Space
- Using the Data Space Manager

## About the Data Space

A region can have a data space, which is controlled by the Data Space Manager. There are various implementation and administrative tasks associated with the data space.

The data space is part of Management Services, and can be used by enabled products, such as NetMaster for TCP/IP and NetMaster for File Transfer (see Figure 18-1).

These products use the data space to hold configuration and connection data. MVS requires that a data space be owned by an address space. For Management Services systems this is the Data Space Manager, the name assigned to the program (and LPAR/region) that owns the data space. The Data Space Manager is started up as a started task and runs for as long as the data space is required. Only one data space is required for each LPAR/region.

For further information, see the *Setting Up Connection Awareness* chapter of the *Unicenter NetMaster Network Management for TCP/IP Administrator Guide*.

*Figure 18-1. How Events and Messages Are Passed to Regions*

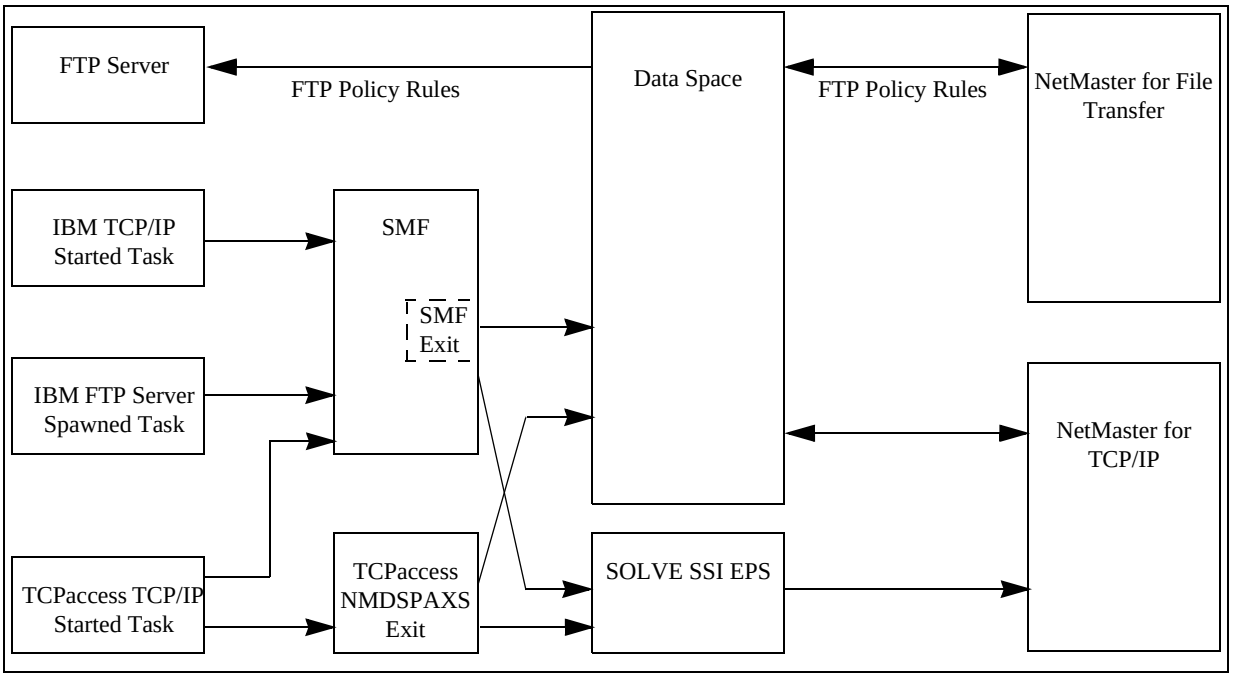


Figure 18-1 shows the following components:

- The data space at the center—this exists as a started task (that is, a region) which runs a program known as the Data Space Manager.
- The SOLVE Subsystem Interface and EPS
- Various TCPaccess and SMF exits that work to populate the data space
- A NetMaster for TCP/IP region with various tasks such as configuration definition, access control for TN3270, and the logging and reporting functions, which all access and update the data space
- A NetMaster for File Transfer region with licensed FTP Server

## Areas of the Data Space

The data space comprises three distinct areas:

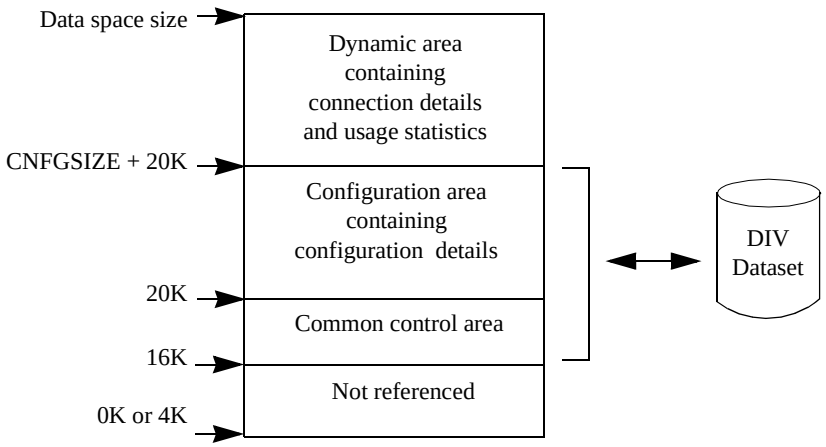
- A common control area which occupies the first 20K of the data space

  **Note**
  The area below 16K is not referenced.

- The configuration area, containing configuration details that are created by using one of the following:
  - The Administration option of NetMaster for TCP/IP
  - The TCPaccess FTP Server option and FTP policy rules of NetMaster for File Transfer
- The dynamic area, containing details of active connections and usage statistics

Data in the common control area (above 16K) and the configuration area is stored in a data-in-virtual (DIV) dataset and loaded by the Data Space Manager into the data space. Figure 18-2 shows the layout of the data space.

*Figure 18-2. Data Space Layout*



The configuration area is divided into 4K blocks, with entire blocks allocated to particular configuration detail record types. The size of the configuration area is set using the Data Space Manager CNFGSIZE parameter. The following table describes the configuration detail record types and can be used to determine the configuration area size which best suits your site's needs. At least one 4K block is allocated to each record type and the minimum configuration size is 40K.

*Table 18-1. Configuration Record Detail Types*

| Class | Record Type | # per 4K Block |
|---|---|---|
| Connection Awareness | Stack Software | 64 |
| Connection Awareness | Application Name | 40 |
| Access Control | Stack | 31 |
| Access Control | Host | 13 |
| Access Control | Port | 13 |
| FTP Management | FTP Policy Rule | 8 |

The size of the dynamic area, and so the total data space size, is controlled by the CONNUM parameter. During initialization, the Data Space Manager allocates a pool of free connection detail records which are used and freed as connections start and terminate. Each 4K block can hold 16 connection details, so any value specified is rounded to the next multiple of 16. In addition, a small amount of storage is set aside for usage statistics. The number of free connections is checked periodically and, if this drops below a threshold (1/8th of the original number of connection records allocated, rounded to the next higher multiple of 16), then the pool is dynamically expanded by the threshold. For example, specifying CONNUM=3000 results in the following:

- Initial allocation of 3008 connection detail records
- A threshold of 384 (3008 / 8 = 376, rounded to 384)

You can use the Data Space Manager DISPLAY command (see the section, *Displaying Data Space Manager Status and Statistics*, on page 18-8) to display information about data space usage.

## About the DIV Dataset

The DIV dataset is used to hold configuration data across IPLs. It must be defined as a VSAM linear dataset. This dataset is allocated by the installation and setup process for your product.

## About the Data Space Manager

The Data Space Manager is a started task that runs a batch program (NMDSPMGR). This program is a long-running job that is started during the IPL process (preferably before the TCP/IP stacks) and never ends. It performs the following actions:

- Allocates the data space
- Populates the data space from the DIV dataset
- Accepts Data Space Manager commands from the console (see below)
- Checkpoints the data space by copying it to the DIV dataset
- Performs periodic housekeeping relating to expired connection details records

Your product's installation and setup process includes a started task, SOLVEDSP, that uses the parameters in the *dsnpref.dsmname*.DSPPARM(DSPSYSIN). The DSPSYSIN member includes the following:

- Dataset allocations (except STEPLIB), for example, the DIV dataset
- Initialization parameters (see the section, *Initialization Parameters*, on page 18-5), specified by using PPREF commands

For further information about this process, see the *Unicenter Mainframe Installation and Setup Instructions*.

## Implementing the Data Space

The Data Space Manager (NMDSPMGR) accepts initialization parameters passed by UTIL0028.

### Initialization Parameters

The Data Space Manager accepts the following initialization parameters:

**DSPNAME=*xxxxxxxx***

A 1- to 8-character identifier for the data space name. If this parameter is omitted, it defaults to SOLVEDSP.

**CNFGSIZE=*nnnnn***

Specifies the amount of storage (in K) that is reserved for the configuration area. The value specified is rounded to the next higher multiple of 4 (K) with a minimum value of 256 (K). Configuration definitions are stored in the configuration area which is copied to the DIV dataset. See the section, *Areas of the Data Space*, on page 18-3 for a description of the configuration area organization.

**CONNUM=*nnnnnn***

Specifies the optimum number of connection records that are recorded in the data space at any time. The value specified is rounded to the next higher multiple of 16. The connection record pool is automatically expanded as described above.

**DELAY=*nn***

Specifies the time (in seconds) to be applied before a data space checkpoint is honored. This can prevent multiple write operations to the DIV dataset when a lot of configuration definitions are added or updated. The default is 60.

**SAFCLASS=*xxxxxxxx***

Specifies the class name of the SAF resource that is used by the Access Control option when validating access by host or port. This is specified as a 1- to 8-character value that is defined to your security system. If this parameter is omitted, a value of FACILITY is used.

**SAFFTP=*xxxxxxxx***

Specifies the class name of the resources that are used by the NetMaster for File Transfer policy control rules when validating access by host or port. This is specified as a 1- to 8-character value that is defined to your security system. If this parameter is omitted, a value of $FTP is used.

**Note**

For further information about customizing TCPaccess FTP Server, see the *Unicenter NetMaster File Transfer Management Implementation, Administration, and Operations Guide*.

**SEC=YES | <u>NO</u>**

Indicates whether access control rules will be processed. The default is NO.

The access control rules are those defined for stacks, ports, and hosts in the administration of the Access Control option of NetMaster for TCP/IP. If you do not intend to use the Access Control option, then you can improve system performance by setting SEC=NO. This parameter does not affect connection awareness.

**SRBTIMER=*n***

Specifies the number of seconds for which an SRB can run in data space code before it is abended. The default is 5.

**TRACE=<u>YES</u> | NO**

Indicates whether data space activity is to be traced. The default is YES.

**TRCSIZE=*nnn***

Specifies (in Kb) the size of the wrap-around trace table. The default is 64.

## Using the Data Space Manager

You can use the Data Space Manager to:

- Display the status and statistics of the data space and verify that the specified CONNUM and CNFGSIZE parameter values are adequate (see *Displaying Data Space Manager Status and Statistics*, on page 18-8)
- Produce dumps to diagnose any problems with the Data Space Manager (see *Producing Dumps as Diagnostic Tools*, on page 18-8)

### Starting the Data Space Manager

To start the Data Space Manager, use the operating system command START (S) as follows:

```
S SOLVEDSP
```

**Note**

As part of your normal IPL procedure, ensure that you start the Data Space Manager before you start your TCP/IP stack. This is because the data space cannot be aware of connections that start before it is running.

### Stopping the Data Space Manager

You can stop the Data Space Manager in any of the following ways:

- By using the operating system STOP (P) command. This command requests confirmation via a WTOR message, and valid responses are Y or N.
- By using the Data Space Manager SHUTDOWN command. This functions exactly like STOP, that is, it requests confirmation.
- By using the Data Space Manager FSTOP command. This causes an immediate termination and there is no WTOR requesting confirmation.

**Note**

Data Space Manager commands are issued by using the operating system MODIFY (F) command; for example:

```
F SOLVEDSP,SHUTDOWN
```

## Displaying Data Space Manager Status and Statistics

Use the Data Space Manager DISPLAY command to show the status of the data space and to verify that the CONNUM and CNFGSIZE parameter values specified are adequate. For example:

```
F SOLVEDSP,DISPLAY
```

where *SOLVEDSP* is the name of the started task.

The DISPLAY command displays Data Space Manager statistics, written as a series of WTOs. Typical output from the DISPLAY command is as follows:

```
N6DQ01 DISPLAY
N6DQ41 CIV  0410 Initialized 2000 101 16.01 Started 2000 101 16.22
N6DQ42 Data space size 0000364K
N6DQ43 Configuration size 0000100K
N6DQ44 Configuration free area size 0000084K
N6DQ45 No. of free connections 000032 of 000032
N6DQ46 Connections expanded by 000016 0000 times
N6DQ47 Automatic connection expansion enabled
N6DQ48 FTP Policy mode is WARN
N6DQ49 FTP Policy ruleset RULESET1 last loaded at 10.30 on 2000/04/10
N6DQ50 FTP Policy ruleset was loaded by USER01 from PRODJOB8
N6DQ03 *END*
```

**Note**

The total number of connections may be slightly more than requested by the CONNUM parameter, because of rounding when obtaining the storage.

## Producing Dumps as Diagnostic Tools

The Data Space Manager provides commands to produce two dumps that may be required as diagnostic tools under instruction from Technical Support. To produce these dumps, use the following commands:

**DSPDUMP**
: Produces a formatted dump of the Data Space Manager.

**DUMP**
: Produces an unformatted snap-dump of the Data Space Manager.

## Formatted Dumps

To produce a formatted dump, issue the Data Space Manager DSPDUMP command in conjunction with the OS/390 MODIFY (F) command; for example:

```
F SOLVEDSP,DSPDUMP
```

where *SOLVEDSP* is the name of the started task.

The output of this dump is written to the *dsnpref.dsmname*.DSPDUMP dataset that was allocated by your product's installation and setup process.

## Unformatted Dumps

To produce an unformatted dump of both the Data Space Manager address space and the data space, issue the Data Space Manager DUMP command in conjunction with the OS/390 MODIFY (F) command; for example:

```
F SOLVEDSP,DUMP
```

where *SOLVEDSP* is the name of the started task.

The output of this dump is written to an OS/390 system dump dataset.

# 19

# Customizing and Using the NetMaster-to-NetSpy Interface

This chapter provides information about customizing and using the interface between NetMaster and NetSpy SNA agents (part of the Unicenter NetSpy Network Performance product).

**This chapter contains the following topics:**

- Managing NetMaster-to-NetSpy Connections
- Managing NetSpy Alerts and Monitors
- Issuing NetSpy Commands
- Using ICS Parameters to Customize the NetMaster-to-NetSpy Interface

## Managing NetMaster-to-NetSpy Connections

NetMaster provides a control for the NetMaster-to-NetSpy (NTN) interface. From this interface you can:

- Activate and inactivate connections between NetMaster and NetSpy SNA agents

  **Note**

  These connections are defined in the NETSPYLINKS parameter group (see the section, *Using ICS Parameters to Customize the NetMaster-to-NetSpy Interface*, on page 19-7).

- Use the console command interface to modify control parameters for NetSpy SNA agents
- Stop the interface to NetSpy SNA agents

To control connections to NetSpy SNA agents, enter **/NASCON** at a ===> prompt. The NetSpy Connections panel is displayed.

*Figure 19-1. NetSpy SNA Agents : NetSpy Connections Panel*

```
PROD------------- NetSpy SNA Agents : NetSpy Connections ---------------------
Command ===>                                                  Scroll ===> CSR

                                        A=Activate I=Inactivate P=Stop F=Modify
   Link Name    ACB Name Status     System   Ver  STC      ITVL
  $ESLNSQ1ND1X  NSQ1ND1N RUNNING    SDD1     6.0  NSQ1ND1  30
  $ESLNSQ1ND1X  NSQ1ND1N RUNNING    SDD1     6.0  NSQ1ND1  30
   **END**
```

This panel displays the status of defined links to NetSpy SNA agents.

For further information about the information displayed and actions available on this panel, press F1 (Help).

## Managing NetSpy Alerts and Monitors

NetMaster receives all alerts from NetSpy. NetMaster alerts are generated for each NetSpy-generated alert received. There are two types of alerts generated:

- Alerts from EPS Services—for general alert monitors defined through NetSpy
- Alerts from the NetMaster-to-NetSpy interface—for user alert monitors defined through NetMaster

### Defining NetSpy User Alert Monitors

Authorized NetMaster users can define, delete, and update NetSpy user alert monitors for a particular resource. These are *ad hoc* monitors.

For further information about these monitors, press F1 (Help) from the NetSpy Monitors List (/NASMON).

To define a NetSpy user alert monitor, do this:

Step 1. Press F4 from the NetSpy Monitors List (/NASMON). The NetSpy Monitor Definition panel is displayed.

*Figure 19-2. NetSpy SNA Agents : NetSpy Commands Panel*

```
PROD----------- NetSpy SNA Agents : NetSpy Monitor Definition -----------------
Command ===>                                                Function=ADD


  Resource Name ...... ____________________
  Monitor Name ....... PRODF7XY
  Monitor Class ...... ____              (VTAM/NCP)
  Resource Type .....+ ________          (Resource/?)
  Match Type ......... ___               (none=ALL/ANY)
  Monitor Interval ... ____              (none=30Sec/Interval)
  LU62 Selection ..... ________          (none/ONLYLU62/EXCLU62)
  Monitor Title ...... ______________________________________________    (no ')

  Threshold Expression
   Variable     Relation Value

 + ________     __       ______
 + ________     __       ______
 + ________     __       ______
 + ________     __       ______
 + ________     __       ______

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

Step 2. Complete the fields on this panel and press F3 to file the monitor definition.

For further information about the input fields on this panel, press F1 (Help).

## Managing NetSpy User Alert Monitors in NetMaster

To manage NetSpy user alert monitors, enter **/NASMON** at a ===> prompt. The NetSpy Monitors List is displayed.

*Figure 19-3. NetSpy SNA Agents : NetSpy Monitors List*

```
PROD------------ NetSpy SNA Agents : NetSpy Monitors List --------------------
Command ===>                                                   Scroll ===> CSR

                        S/B=Browse U=Update C=Copy D=Delete A=Activate Q=Query
   Resource Name        Monitor Name   Status      Blocks Rcvd    NetSpy Link
   NCP1601              PROD647S       DEFINED            1567    $ESLQA1SPYN1
   SSTCP034             PRODOIQI       DEFINED            1531    $ESLQA1SPYN1
   **END**
```

This panel lists the NetSpy user alert monitors defined for a resource.

For further information about the information displayed and actions available on this panel, press F1 (Help).

## Issuing NetSpy Commands

NetMaster supports a NetSpy command interface. This allows a subset of display commands to return information to NetMaster.

To use the NetSpy Agent Commands function, enter **/NASCMD** at a ===> prompt. The NetSpy Commands panel is displayed.

*Figure 19-4. NetSpy SNA Agents : NetSpy Commands Panel*

```
PROD---------------- NetSpy SNA Agents : NetSpy Commands -----------------------
Command ===>                                                  Scroll ===> CSR

                                                                        S=Select
    Command    Action
    DA       - Display Application Traffic and Response Time
    DAR      - Display Application Response Time Distribution
    DN       - Display NCP Statistics
    DNETH    - Display Ethernet Network Statistics
    DNFR     - Display Frame Relay Network Statistics
    DNLIST   - Display NCP List
    DNS      - Display NCP Node Monitoring Status
    DNTIC3   - Display TIC3 Network Statistics
    DNTR     - Display Token Ring Network Statistics
    DNX25    - Display X.25 Network Statistics
    DT       - Display Terminal Traffic and Response Time
    DTA      - Display Terminal Response Time Analysis
    DTR      - Display Terminal Response Time Distribution
    DV       - Display Network delays for Virtual Routes
    **END**


 F1=Help      F2=Split     F3=Exit                   F5=Find
 F7=Backward  F8=Forward   F9=Swap
```

This panel lists the NetSpy SNA agent commands that you can issue. To issue a command, do this:

Step 1. Enter **S** beside the command. The NetSpy Command Arguments panel is displayed.

*Figure 19-5. NetSpy Agent Services : NetSpy Command Arguments Panel*

```
PROD-------------- NetSpy SNA Agents : NetSpy Command Arguments ---------------
Command ===>                                              Function=ACTION

    Command ............ DT

    LUName ............. ________________  (none=ALL)
    APPL ............... ________
    Mode ............... ________          (none=INTERVAL|SUMMARY|DELTA)
    LU62 Selection ..... ________          (none|ONLYLU62|EXCLU62)

  Threshold Expression
  Variable      Relation Value
 + ______         > __________
```

Step 2. Type values in the fields for any operands that you want to use.

Step 3. Press F6 (Action) to issue the command. The command output is displayed.

*Figure 19-6. NetSpy SNA Agents : NetSpy Command Results Panel*

```
PROD-------------- NetSpy SNA Agents : NetSpy Command Results (DT)-----------------
Command ===>                                                          Scroll=CSR
******************************** TOP OF DATA ***********************************
12:20 ** RESPONSE TIME STATISTICS (INTERVAL) **         WED 26-SEP-2001 11.12.20
SECONDARY   AVG-RESP    WORST-RESP     NO.     NO.       NO.  OUTPUT  THINK  SESS
LU NAME    HOST   NET   HOST   NET   TRANS.  INPUT    OUTPUT    WAIT   WAIT  TYPE
------------------------ APPL NAME=PROD1   -------------------------------------
 SSTCP043   0.7   0.0    5.2   0.0       74     74        80     0.0   41.8
 SSTCP048   0.0   0.0    0.0   0.0        0      0         0     0.0
 SSTCP047   0.0   0.0    0.0   0.0        0      0         0     0.0
 SSTCP041  16.4   0.0   55.3   0.0        6      5         7     0.0   9.1M
 SSTCP042   0.3   0.0    2.4   0.0       38     38        52     0.0   2.9M
 SSTCP044   0.5   0.0    5.4   0.0       35     35        60     0.0    1.8
-------------------------------------------------------------------------------
12:16 ** RESPONSE TIME STATISTICS (INTERVAL) **         WED 26-SEP-2001 11.12.16
SECONDARY   AVG-RESP    WORST-RESP     NO.     NO.       NO.  OUTPUT  THINK  SESS
LU NAME    HOST   NET   HOST   NET   TRANS.  INPUT    OUTPUT    WAIT   WAIT  TYPE
------------------------ APPL NAME=PROD1   -------------------------------------
 SSTCP043   0.7   0.0    5.2   0.0       74     74        80     0.0   38.0
 SSTCP048   0.0   0.0    0.0   0.0       74     74         0     0.0
 SSTCP047   0.0   0.0    0.0   0.0       74     74         0     0.0
 SSTCP041  16.4   0.0   55.3   0.0        6      5         7     0.0   9.0M
 SSTCP042   0.3   0.0    2.4   0.0       38     38        42     0.0   2.8M
 SSTCP044   0.5   0.0    5.4   0.0       34     34        58     0.0    1.6
****************************** BOTTOM OF DATA **********************************
```

**Note**

You can also issue a command, together with any operands, by entering it directly at the Command ===> prompt on the NetSpy Commands panel. If you enter issue a command without any operands, it is issued with its default operands.

## Using ICS Parameters to Customize the NetMaster-to-NetSpy Interface

The NTN interface is implemented during initialization of your NetSpy Network Performance product. To customize the interface after initialization, update the NETSPYLINKS parameter group in Initialization and Customization Services (ICS).

To update the NETSPYLINKS parameter group, do this:

Step 1. Enter **/ICS** at the ===> prompt. The ICS : Customization Parameters list is displayed.

Step 2. Enter **U** beside the NETSPYLINKS parameter group. The Initialization Parameters panel for the NETSPYLINKS parameter group is displayed.

*Figure 19-7. The NETSPYLINKS Parameter Group*

```
PROD------------------ ICS : Initialization Parameters -------------Page 1 of 1
Command ===>                                                  Function=Update

.- NETSPYLINKS - Links to NetSpy SNA Agents ----------------------------------.
|                                                                              |
| NetSpy SNA Agent Connections:                                                |
|    NetSpy Application ID 1 ........           (NSYXname of NetSpy)           |
|    NetSpy Application ID 2 ........             ''      ''   ''              |
|    NetSpy Application ID 3 ........             ''      ''   ''              |
|    NetSpy Application ID 4 ........             ''      ''   ''              |
|    NetSpy Application ID 5 ........             ''      ''   ''              |
|    NetSpy Application ID 6 ........             ''      ''   ''              |
|    NetSpy Application ID 7 ........             ''      ''   ''              |
|                                                                              |
| Reconnect Retry Interval .......... 120       (10 to 600 seconds)            |
| Solicited Response Timeout ........ 30        (5 to 120 seconds)             |
| Enable NetSpy Alert Processing .... YES       (default is YES)               |
| Default LOGMODE ................... INTERACT  (default is INTERACT)         |
'------------------------------------------------------------------------------'
.- Notes ---------------------------------------------------------------------'
| Only identify the NetSpy SNA Agent applications running on this system/LPAR |
'------------------------------------------------------------------------------'
 F1=Help      F2=Split     F3=File      F4=Save      F5=ILog      F6=Action
                           F9=Swap                                F12=Cancel
```

Step 3. In the Connections fields specify the values of the NSYXNAME parameter in the INITPRM member of the NetSpy SNA agents that are to be linked to your region.

Step 4. Enter a value in each field that you require.

For further information about completing this panel, press F1 (Help).

Step 5. Press F6 (Action) to set the changes; press F3 (File) to save the changes.

**Note**

The Enable NetSpy Alert Processing field in the NETSPYLINKS parameter group allows you to switch off the receipt of all alerts from NetSpy SNA agents. Normally, you will leave the field to its default value of YES. However, you might want to enter **NO** to switch the alerts off under abnormal conditions (for example, when the region is flooded by these alerts).

# A

# Connecting Terminals to Management Services

This appendix provides information about connecting terminals to Management Services.

**This chapter contains the following topics:**

- Supported Terminals
- How Management Services Accepts Terminal Connections
- Extended Attributes Support
- Specifying Screen Sizes
- Connecting Hitachi 560/20 Devices
- Specifying Logmode Table Definitions

## Supported Terminals

The IBM devices listed below are supported by Management Services. Any device compatible with those listed is also supported.

| IBM Device or Equivalent | Supported Functions |
|---|---|
| 3270 Family LU0, LU1 (up to 255 x 255) | Full function |
| 5550 (24 x 80) | Full function, including DBCS |
| 3767 (or other LU-1 device) | EASINET, OCS, printing |

In addition, color, extended highlighting and DBCS datastreams are supported on the relevant Fujitsu terminals.

The following Hitachi devices are supported by Management Services on VOS3:

| Hitachi Device | Supported Functions |
|---|---|
| T-560/20 display terminal (24 x 80) | Full function |

Information that allows Management Services to connect to these terminals is contained in their VTAM mode table definition. See the section, *Specifying Logmode Table Definitions*, on page A-6, for information on the expected parameter settings for different terminal types.

**Note**

Management Services 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. See Chapter 3, *Implementing Security*, for more information about defining a system console.

## How Management Services Accepts Terminal Connections

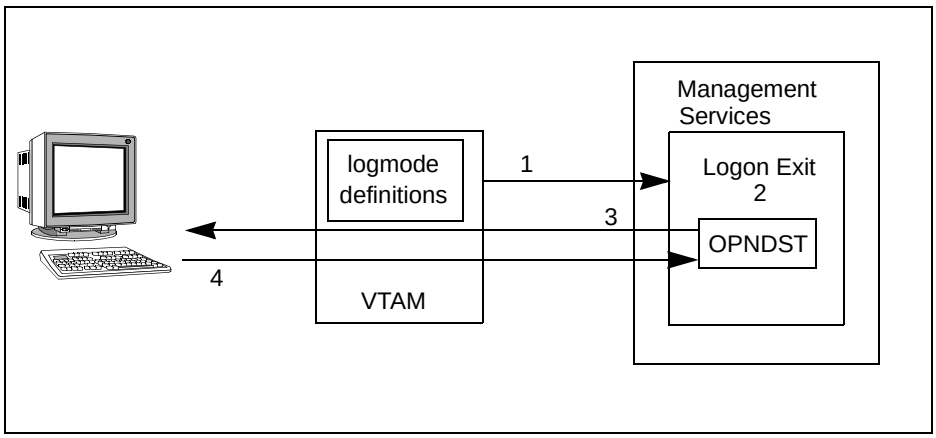
When a request for a terminal connection is made to Management Services, the following process occurs:

1. The logon exit is driven
2. The logon exit examines the session parameters to determine the terminal type

   See the section, *Specifying Logmode Table Definitions*, on page A-6, for information on the parameters that are expected for different terminal types.
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.

Figure A-1 shows what happens when a terminal connection is accepted.

*Figure A-1. Successful Terminal Connection*



## 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 Management Services 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 datastreams for special language support)
- Screen size

A query is sent by Management Services when X‘80’ is specified in the second position of the PSERVIC field of the logmode definition, for example, PSERVIC=X‘0280...’.

## Specifying 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.

To allow all terminals in a network to support unspecified viewport size regardless of how the screen sizes have been coded in the mode table definition, use the SYSPARMS UNSPVPSZ operand. If set to YES, Management Services dynamically modifies the session parameters for the terminal to specify unspecified viewport size. All terminals can then run at the largest screen size of which it is capable.

**Note**

Management Services does not alter the session parameters 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.

## Connecting Hitachi 560/20 Devices

On a VOS3 system, Management Services supports connection to both 3270 and 560/20 terminal devices. When a terminal logs on, its device type is identified as follows:

- The table of static terminal definitions built using the DEFTERM command is scanned for a definition matching this LU name
- If no definition is found, the terminal type is determined by a query. If the eleventh position in the PSERVIC field is set to X‘40’ the terminal is assumed to be a 560/20. Otherwise it is treated as a 3270.

**Note**

If the terminal is incorrectly identified, the wrong data format is used and a terminal check results.

Management Services supports extended attributes for 560/20 devices. A fixed screen size of 24X80 is used and terminal attributes such as color and DBCS capability can be supplied using the DEFTERM command.

It is recommended that the Hitachi default logmode HNA563V be used by 560/20 devices connecting to Management Services.

## Specifying Logmode Table Definitions

The following table specifies the logmode table parameter values that are expected by Management Services for different terminal types:

| Terminal Type | Expected Parameter Values |
|---|---|
| LU01 | FMPROF=X‘02’ |
| LU02 | FMPROF=X‘03’<br>PSERVIC=X‘01...’ |
| LU03 | FMPROF=X‘03’<br>PSERVIC=X‘02...’ |

To help systems programmers when installing Management Services, examples of suggested mode table entries for most 3270 screens can be found in the MODETABS member in the base installation library BASE.INSTALL. 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 EASINET is also supplied in the member.

For more information on coding mode table entries, see the *VTAM Resource Definition Reference* manual (SC31-6552-00).

# B

# JCL Parameters

This appendix lists the Management Services JCL parameters and provides a description of each.

## JCL Parameter Summary Table

Table B-1 provides a summary of the available JCL parameters. See the following section for more detailed information on how to specify each parameter.

*Table B-1. JCL Parameter Summary Table*

| JCL Parameter | Description |
|---|---|
| AM | Specifies the access method to be used by Management Services for communication between terminals. |
| ARMNAME | Specifies the name used to register a region with the Automatic Restart Manager (ARM). |
| CNM | Specifies the name of the VTAM CNM ACB APPL which the NEWS and NTS option components use. |
| DBCS | Specifies the DBCS option. |
| DSNQLCL | Specifies the local dataset qualifier. |
| DSNQSHR | Specifies the shared dataset qualifier. |
| DYNVVOL | Specifies a default volume for dynamically allocating files. |

*Table B-1. JCL Parameter Summary Table*

| JCL Parameter | Description |
|---|---|
| INIFILE | Specifies the INI file to use for customization of parameters. |
| INIRESET | Indicates that all current parameters are to be ignored and the defaults used. |
| INIT | Specifies the name of the NCL procedure residing in the COMMANDS procedure library that is executed as part of system initialization, before VTAM ACBs are opened. |
| INIWTO | Indicates that initialization messages are to be displayed on the system console. |
| INT | Determines which internal commands can be used. |
| NCLFM | Specifies the default CMS FILEMODE that applies to loading NCL procedures. |
| NCLFT | Specifies the default CMS FILETYPE that is to apply during the loading of NCL procedures, where no overriding FILETYPE is provided in the connecting user's UAMS definition. |
| NMDID | Specifies the domain ID for this system. |
| NMSUP | Specifies the System User Prefix for background user IDs. |
| NOMODIFY | Specifies that operating system MODIFY facility communication with the product via the system console is not supported. |
| NPF | Specifies whether NPF is used on the system. |
| NPFFM | Specifies that default CMS FILEMODE applies when loading NPF tables. |
| NPFFT | Specifies that default CMS FILETYPE applies when loading NPF tables. |
| OPT | Activates CPU accounting support. |
| OSINP | Controls access to the OSCNTL dataset. Access can be either read-only or update. |
| PPO | Specifies the name of the primary program operator (PPO) ACB APPL to be used for PPO messages requested from VTAM using the PPO START command. |
| PRI | Specifies the name of the primary VTAM ACB APPL to be used for terminal communication. |
| PROD | Specifies the list of features to be included in the system. |
| READY | Specifies the name of the NCL procedure within the COMMANDS procedure library that executes as part of system initialization after VTAM ACBs have been successfully opened. |

*Table B-1. JCL Parameter Summary Table*

| JCL Parameter | Description |
|---|---|
| SEC | Specifies whether a security exit is to be used. |
| SSID | Specifies whether a connection is to be made to the subsystem interface (SSI) during Management Services initialization. |
| TZ | Specifies whether the internal timezone of Management Services is set to Greenwich Mean Time or system 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 (OS/VS systems only). |
| VFSENQ | Specifies whether the VFS dataset is enqueued exclusively on initialization. |
| VSAMIO | Specifies when VSAM I/O is performed in the Management Services main task, or via a separate subtask. |
| WTO | Specifies whether the system console is to be sent monitor class messages. |
| XOPT | Specifies options for dump processing and ARM registration. |

## JCL Parameter Descriptions

This section provides detailed descriptions of all the JCL parameters listed in the summary table.

**AM={ VTAM | XNF }**

Specifies the access method to be used by Management Services for terminal communication. Valid values are VTAM and XNF. The default is VTAM. XNF is valid only for Hitachi VOS3 systems.

**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.

**CNM={ NMCNM | *name* }**

Specifies the name of the VTAM CNM ACB APPL which the NEWS and NTS components use. If specifying this parameter, the name must be 1 to 8 characters, which defaults to NMCNM.

**DBCS={ YES | NO | IBM | FUJITSU }**

Specifies the DBCS option.

**DYNVVOL=*value***

Specifies a default volume for dynamically allocating files. If this parameter is not specified, then no volume is specified in the dynamic allocations and SMS allocates the files according to its rules.

**DSNQLCL**

Specifies the name of the local dataset qualifier. This value is used when allocating datasets.

**DSNQSHR**

Specifies the name of the shared dataset qualifier. This value is used when allocating datasets.

**INIFILE=*filename***

Specifies the INI file to use for customization of parameters.

**INIRESET**

Indicates that all current parameters are to be ignored and the defaults used. This has no effect if an INI file is specified.

**INIT={ RMINIT | *name* }**

Specifies the name of the NCL procedure, residing in the COMMANDS procedure library, that is executed as part of system initialization before VTAM ACBs are opened. The name must be 1 to 8 characters. The default is RMINIT.

**INIWTO**

Indicates that initialization messages are to be 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 Table B-2. The recommended value is INT=E4.

*Table B-2.Values of INT Parameter*

| 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* }  (for VM systems only)**

Specifies the CMS filemode that applies to loading NCL procedures. This must be a valid CMS filemode. The default is an asterisk (*).

**NCLFT={ NCL | *filetype* }  (for VM systems only)**

Specifies the CMS filetype that is to apply during the loading of NCL procedures, where no overriding filetype is provided in the connecting user's UAMS definition. The filetype must be a 1- to 8-character name. The default is NCL.

**NMDID=*domainid***

Specifies the domain ID for this system. The domain ID must be a 1- to 4-character name. If the domain ID is not specified, the first characters (up to a maximum of 4) of the system primary ACBNAME, 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 Remote Operator Facility (ROF) sessions from background environments. Do not confuse this parameter with the long name of the system, as described in the SYSPARMS ID parameter.

**NMSUP=*userprefix***

Specifies the System User Prefix. This prefix is a 1- to 4-character name that is used to prefix the user IDs for background system environments. If this prefix is not specified, the domain ID (NMDID) is used. See NMDID above for details of what happens if the domain ID is not specified.

**NOMODIFY  (for OS/VS systems only)**

Specifies that communication between the operating system MODIFY facility and the product, via the system console, is not supported. If omitted, the use of MODIFY is supported.

**NPF={ NO | YES }**

Controls the use of NPF. Specify NO to inhibit NPF on the system. This is useful on test systems where security checking is not required but where a production UAMS is used.

**NPFFM={ * | *a* } (for VM systems only)**

Specifies that the CMS filemode applies when loading NPF tables. A valid CMS filemode must be specified. The default is an asterisk (*).

**NPFFT={ NPF | *filetype* } (for VM systems only)**

Specifies that the CMS filetype applies when loading NPF tables. The filetype must be a 1- to 8-character name. The default is NPF.

**OPT=00/*nn***

Activates CPU accounting support in the dispatcher. The parameter value comprises one hexadecimal byte, where each bit indicates the level of accounting, as in Table B-3.

*Table B-3.Values of 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 dataset. Can be either read-only or update. Specifying YES (the default) indicates that the dataset 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, Management Services retries the open for read only access.

**PPO={ NMPPO | *ppoacb* }**

Specifies the name of the primary program operator (PPO) ACB APPL to be used for PPO messages requested from VTAM using the PPO START command. The name must be 1 to 8 characters. The default is NMPPO.

**PRI={ NM | *acbname* }**

Specifies the system name of the primary VTAM ACB APPL name to be used for communication. The name must be 1 to 8 characters. The default is NM.

It is recommended that this parameter (rather than the SYSPARMS PRIACBNM command in RMINIT) be used to set the ACB name, as some system components use the name as set in the PRI JCL parameter and do not recognize or allow changes to the value later.

**Note**

If this parameter is omitted, the default (NM) is also the default for the following parameters:

- NMDID
- NMSUP (if NMDID is not specified)
- SYSPARMS ID

**PROD={ *product* | (*product1,product2,...productn*) }**

Specifies the list of product name key values for features to be included in the region. The PROD list, for example, PROD=(SNA,TCPIP), 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.

Table B-4 shows product names with their associated product name keys and LMP codes for OS/390 and z/OS.

**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 italicized. 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. For example, if you are licensed for File Transfer (FT), you may want FTP support but not CONNECT:Direct support.

**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. For example, PROD=TCPIPACNTL requires PROD=TCPIP. Note that you can specify these in any order.

*Table B-4. Product Name Key Values for PROD= on OS/390 and z/OS*

| Product Name | Product Name Key | LMP Code |
|---|---|---|
| NetMaster for SNA | SNA | YX |
| NetMaster Automation | SNAAUTO | XY |
| – *NetView Connect* | NVC | — |
| NetMaster for TCP/IP | TCPIP | Y7 |
| – *NetMaster for TCP/IP Access Control* | TCPIPACNTL | — |
| NetSpy (NetMaster for TCP/IP Performance) | NETSPY | DY |
| NetMaster Network Operations for TCP/IP | NETOPER | 2T |
| NetMaster for File Transfer | FT | X3 |
| – *CONNECT:Direct Support* | FTCD | — |
| – *FTS Support* | FTFTS | — |
| – *FTP Support* | FTFTP | — |
| – *Mailbox Support* | FTMBX | — |
| SOLVE:Access | SNAACCESS | ZB |
| SOLVE:FTS | FTS | ZI |
| SOLVE:Link forDB2 (EDBS) | EDBS | ZK |
| INFO/MASTER | INFO/MASTER | 09 |
| NET/MAIL | NET/MAIL | 1K |

*Table B-4. Product Name Key Values for PROD= on OS/390 and z/OS*

| Product Name | Product Name Key | LMP Code |
|---|---|---|
| SOLVE:Central - Asset | ASSET | ZR |
| SOLVE:Central - Change | CHANGE | ZT |
| SOLVE:Central - Configuration | CONFIG | ZU |
| SOLVE:Central - Problem | PROBLEM | ZV |
| SOLVE:Operations for OS/390 | OPSOS | ZX |
| SOLVE:Operations for CICS | OPSICS | Z5 |

**READY={ RMREADY | *membername* }**

Specifies the name of the NCL procedure, within the COMMANDS procedure library, that executes as part of system initialization after VTAM ACBs have been successfully opened. The name must be 1 to 8 characters. The default is RMREADY.

**SEC={ * | NO | PARTSAF | *name* }**

Indicates if a security exit is to be used by this region.

If an asterisk (*) is specified (the default), then the region uses a security exit if one has been linkedited into the NM001 load module. If no security exit has been linkedited, then the region uses the NMUEX01 load module if it is in a Management Services-accessible load library.

If NO is specified, then no security exit is used. This overrides any linkedited exit or the NMUEX01 load module.

If PARTSAF is specified, then a vendor-supplied partial security exit that uses SAF is used.

If *name* is specified, then the named load module is loaded and used as the security exit. If this load module cannot be found, then the region terminates.

**Warning**

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.

See Chapter 4, *Implementing Additional Security*, and Appendix F, *User ID Security Exit Support*, for additional information on security exits.

**SSID={ NO | * | *name* }  (for OS/VS systems only)**

Used to establish a connection to the subsystem interface (SSI) during Management Services initialization.

If NO (the default) is specified, then no connection to an NMSSI is attempted. The connection is only started (or attempted) after a SYSPARMS SSID command is issued.

If an asterisk (*) is specified, then an SSID of the first four characters of the Management Services 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 default value of the SYSPARMS SSIRTINT operand.

**TZ={ *sHHMM* | GMT }**

Sets the internal timezone of Management Services. Do not specify this parameter if you want Management Services to run on system local time.

The system hardware clock is usually set to Greenwich Mean Time (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, the TZ parameter can be used to reset this time. *s* is plus (+) or minus (-), depending on whether the site is ahead of (+) or behind (-) GMT. *HHMM* is the four-digit hours and minutes value of the time zone offset.

If GMT is specified, then, assuming that the hardware clock is set to GMT, Management Services sets its internal time to GMT, rather than system local time.

**UDBDEFER  (By default on OS/VS systems only)**

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 Local Shared Resource (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 }  (OS/VS systems only)**

Specifies whether the VFS dataset 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 Management Services started task, MVS does 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 datasets might fail to open if SHROPTIONS 1 or 2 are in effect.

**VSAMIO={ M | S | D }**

Specifies when VSAM I/O is performed in the Management Services main task. 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 (normally *pppp*OPER). The default is YES.

**XOPT={ *option* | *option*, *option*, ... }**

Specifies options for dump processing and ARM registration. This operand can be specified more than once. The following keywords are valid:

| | |
|---|---|
| SDUMP | Write ABEND dumps to SYS1.DUMP dataset. |
| NOSDUMP | Send ABEND dumps to normal dump datasets (default before starting parm processing). |
| DAE | Provide DAE symptom information when writing an ABEND dump. |
| NODAE | Do not provide DAE symptoms (default before starting parm processing). |
| ARM | The region attempts to register with the Automatic Restart Manager (ARM), using an ARM element name as specified. |
| NOARM | The region does not register with ARM (this is the default). |

Use the ARMNAME= JCL parameter to register a region with ARM. The sysplex automatic restart manager restarts a registered region automatically if that region fails.

# C

# The UTIL0028 Utility

This appendix provides information about the UTIL0028 utility program and how to use it.

## About UTIL0028

UTIL0028 is a utility program that is executed by the region started task in an MVS (any variant), MSP, or VOS3 environment.

The execution JCL executes the UTIL0028 program as part of the installation process for Unicenter, NetSpy, and SOLVE products. UTIL0028 reads the *dsnpref*.SOLVE.JCL (RUNSYSIN) member which contains all the parameters necessary to allocate datasets and SYSOUT files to invoke Management Services. RUNSYSIN is generated as part of the installation process and can be customized.

UTIL0028 also allows you to supply additional JCL parameters to overcome the OS 100-character JCL parameter limit. (JCL parameters are documented in Appendix B, *JCL Parameters*.)

When starting a region before JES, you can use this facility to avoid the need to catalog Management Services datasets in the master catalog.

If using UTIL0028 on an MVS/ESA 5.2 or later system, you can use the MVS system substitution service to substitute data in the UTIL0028 control file. This aids in cloning systems in a SYSPLEX environment.

## Using UTIL0028

When UTIL0028 executes, it points to the U28SYSIN dataset to provide all the parameters required to allocate dataset names and then invoke your product. The following section describes how to specify the contents of U28SYSIN.

### Specifying the U28SYSIN Dataset

The U28SYSIN DD statement points to the dataset that contains control statements for UTIL0028. These control statements allow:

- The specification of UTIL0028 options
- PARM information for the target program
- Dataset or SYSOUT allocation requests

The attributes specified in the dataset must be:

- Fixed (Blocked or Unblocked)
- LRECL 80
- Any valid block size

The dataset can be sequential or a 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 UPPER case.
- Only columns 1 to 72 are examined. If line numbers are desired, they can be in columns 73 to 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 UTIL0028 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, provided that they are separated from the commands and operands by a blank.
- The commands for dataset and SYSOUT allocation might need to 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.

## UTIL0028 Commands

There are three groups of UTIL0028 commands:

- General
- The dynamic allocation of datasets
- The dynamic allocation of SYSOUT datasets

### General Commands

The following general commands can be specified in the U28SYSIN dataset:

**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***

Return to the system with *nnnn* return code

**U*nnnn***

Abend with user abend code *nnnn*. If a code is not specified, it defaults to 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). It is not used between individual sections of the prefix and/or suffix.

The default value is a comma (,).

**PPREF=*value***

Specifies the prefix of the supplied JCL parm, 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.

**PSUFF=*value***

Specifies the suffix of the supplied JCL parm 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.

**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 (see below), 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'***

This statement allows you to define up to 20 additional user variables for substitution. *xxx* is the variable name and the value is specified after the equals sign (=) which is 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). This is a rule enforced by the symbol substitution service to prevent buffer overruns.

## Dynamically Allocating a Dataset

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

```
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, an operand (by itself) or plus sign (+) indicates 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 dataset is to be dynamically concatenated to the previous dataset allocation (which must be a dataset, not a SYSOUT allocation).

**DSN=*datasetname***
Specifies the dataset to be allocated. A PDS member name or GDG relative generation number can be specified in parenthesis after the dataset name.

**DISP=( { OLD | SHR | MOD | NEW } [ ,*ndisp* [ ,*cdisp* ] ] )**
Specifies the dataset 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 dataset is (to be) on. For existing datasets, bypasses a catalog search. Generally requires that the UNIT operand also be specified.

**UNIT=*unitname***
Specifies the unit type that the dataset is (to be) on. This is required for existing datasets if the volume is specified.

**CYL=( *pri,sec* ) | TRK=( *pri,sec* )**
For new datasets, allows specification of a space allocation value. Note that you cannot allocate a PDS, as no directory amount can be specified.

**BLKSIZE=*nnnnn***
Lets you specify a block size. This is also useful when concatenating datasets of unlike block size, as it also sets the DCB BUFLEN value.

## Dynamically Allocating a SYSOUT Dataset

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

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

If the operands will not fit on one line, an operand, by itself, or plus sign (+), indicates that the command is continued. For example:

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

The operands to dynamically allocate a SYSOUT dataset 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={ <u>CLOSE</u> | 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. For Management Services, you *should not* use FREE=CLOSE for FMTDUMP.

**HOLD={ <u>NO</u> | YES }**

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

**BLKSIZE=*nnnnn***

Lets you specify a block size.

## UTIL0028 Processing

After performing any requested dynamic allocation, and prefixing and/or suffixing the supplied JCL PARM, UTIL0028 transfers control (using XCTL) to a nominated program (by default, NM001).

UTIL0028 processes in the following sequence:

1. The U28SYSIN dataset is opened
2. Input commands are read and processed
3. The U28SYSIN dataset is closed
4. If no errors are detected, the target program is XCTL'd to, or passed the input JCL PARM, or prefixed, or suffixed, as requested.

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.

**Note**

If reassembling UTIL0028, the assembler SET symbol, &MVS520 must be set appropriately to enable (1) or disable (0) the assembler code related to the MVS symbolic substitution service support (the SUBS=YES option).

If you do not set this variable to 0 in a non-MVS/ESA 5.2 or later system, you will get assembly errors.

# D

# SYSPARMS Operands

This appendix lists all the operands for the SYSPARMS command and provides detailed information on their use.

**This appendix contains the following topics:**

- Generic SYSPARMS Summary Table
- Generic SYSPARMS Operand Descriptions
- SYSPARMS Operands for NEWS
- SYSPARMS Operands for NTS
- SYSPARMS Operands for SYSCMD

## Generic SYSPARMS Summary Table

Table D-1 provides a summary of all the available SYSPARMS operands. See the following section for details on specifying each operand.

*Table D-1. SYSPARMS Summary Table*

| SYSPARM | Description | See page |
|---|---|---|
| ABENDCD | Specifies the user abend code for use when Management Services ABENDs (OS/VS and VM systems only). | D-12 |
| ACBRETRY | Specifies whether Management Services is to attempt to re-open the VTAM ACB. | D-12 |
| APPLSTAT | Specifies the name of a VTAM APPL definition for which automatic status checking is performed. | D-12 |
| APPLSTIV | Specifies the time interval at which the status of an application is checked. | D-13 |
| APPTXT1-6 | Assigns a word which describes the status of an application monitored by APPLSTAT processing when the application has a corresponding VTAM status of: ACTIVE, INACTIVE, NOLOGONS, UNKNOWN, UNAVAIL, or NOTAPPL. | D-13 |
| AUTOEXEC | Specifies whether Management Services attempts to execute an unrecognized command string as an NCL procedure. | D-14 |
| CALLSHR0 | Specifies whether a subtask shares subpool zero with the main task when an &CALL statement is executed (OS/VS and VM systems only). | D-15 |
| CDELAY | Specifies the time Management Services waits before sending output to an OCS terminal when a user is entering input from the keyboard. | D-15 |
| CMDAUTH | Changes the command authority for the specified command. | D-15 |
| CMDREPL | Specifies an NCL procedure to be run in the place of a specified command. | D-16 |
| CMDSTACK | Specifies the size of the OCS command stack. | D-16 |
| CONMSG | Specifies whether the message N07002 is written to the activity log each time a terminal connects to the system. | D-17 |
| CONUIDPW | Specifies whether Management Services is to accept user data when a user logs on to the system. | D-17 |
| DALDEFER | Specifies whether deferred mounting is to be requested when allocating datasets (OS/VS systems only). | D-17 |
| DALRACF | Specifies whether automatic RACF protection is requested when dynamically allocating new datasets (OS/VS systems only). | D-17 |

*Table D-1. SYSPARMS Summary Table*

| SYSPARM | Description | See page |
|---|---|---|
| DALRLSE | Specifies whether datasets created by dynamic allocation are defined with the RLSE option (OS/VS systems only). | D-17 |
| DBCS | Specifies whether the system is to support devices that are capable of operating with DBCS. | D-18 |
| DESC | Specifies the operating system description codes used for messages sent to the system console (OS/VS systems only). | D-18 |
| DYNLMAX | Specifies the maximum number of dynamic INMC links that can be concurrently active. | D-18 |
| EASINET | Specifies whether EASINET is to be active. | D-18 |
| EDITCAPS | Specifies the default setting for the CAPS command. | D-18 |
| EDITMAXK | Specifies the maximum amount of storage the system can allow for the Management Services editor. | D-19 |
| EDITNULL | Specifies the default setting for the NULLS command. | D-19 |
| EVCMDMIN | Specifies the minimum repeat frequency for the EVERY command. | D-19 |
| GENRSRC | Specifies the VTAM generic resource support name to be used during initialization of your region. This is used if you want to implement SOLVE:Access as a generic resource in a sysplex. | D-19 |
| 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. | D-20 |
| ID | Specifies the system identifier. | D-20 |
| INMCBFSZ | Specifies the INMC buffer size for INMC traffic. | D-20 |
| INMCEX01 | Defines the load module or phase name for the INMC primary security exit. | D-20 |
| INMCEX02 | Defines the load module or phase name for the INMC secondary security exit. | D-21 |
| IPAMHB | Controls the use of heartbeats for TCP/IP INMC and APPC links. | D-21 |
| IPCHECK | Associates IP addresses with user IDs for logon processing and checking. | D-21 |
| IPPORT | Specifies information to enable the sockets interface to IBM TCP/IP or TCPaccess. | D-22 |

*Table D-1. SYSPARMS Summary Table*

| SYSPARM | Description | See page |
|---|---|---|
| JRNLPROC | Specifies the NCL procedure which is to be started when a journal swap occurs. | D-22 |
| JRNLSWAP | Indicates whether the NDB journal dataset is to be automatically swapped if a space error occurs on the active journal. | D-22 |
| LANG | Specifies the language code for the system. | D-22 |
| LMSGWARN | Specifies the repeat frequency at which OCS operators are warned of lost messages. | D-22 |
| LNKTRACE | Specifies whether a trace message is to be issued each time an attempt to open a session to a remote region fails. | D-22 |
| LOCKMAX | Specifies the maximum number of &LOCK requests that can be concurrently active. | D-23 |
| LOCKPROC | Specifies the procedure that is invoked when a LOCK command is issued. | D-23 |
| LOGPAGE | Specifies the number of lines the activity log has on each page. | D-23 |
| LOGPROC | Specifies the NCL system procedure that is to be used to intercept and process all messages issued to the activity log. | D-23 |
| LOGWRAP | Specifies whether the log datasets can be reused. | D-23 |
| LU1BKSP | Specifies what to do with the backspace character when reading data from an LU1 device. | D-23 |
| MAIACBOR | Specifies the maximum number of retries allowed when opening an MAI ACB. | D-23 |
| MAIEX02 | Specifies the name of an exit routine to take control whenever an MAI-OC session is started or ended. | D-24 |
| MAIEX02S | Specifies the way in which MAI-OC serializes calls to the MAIEX02 exit routine. | D-24 |
| MAIFILE | Specifies the UDB name of the MAI session definition file (non-VFS). | D-24 |
| MAIONL | 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). | D-25 |
| MAIOPREF | Specifies a 1- to 5-character string which is used as the prefix to an LU name generated by MAI-OC. | D-25 |
| MAIOTRNS | Specifies the translate table used by MAI-OC. | D-25 |

*Table D-1. SYSPARMS Summary Table*

| SYSPARM | Description | See page |
|---|---|---|
| MAPDEL | Removes a loaded mapping services map from memory. | D-25 |
| MAPFILE | Specifies the file used by mapping services to store and load maps. | D-25 |
| MAPLOAD | Loads a mapping services map into memory. | D-25 |
| MAPRESET | Queues a mapping services map for removal from memory and loads a fresh version of the map. | D-26 |
| MAXPANEL | Specifies the number of panels held in storage. | D-26 |
| MAXRUSZ | Specifies the maximum request unit size for APPC sessions. | D-26 |
| MAXRUT2 | Specifies the maximum request unit size used when sending chained output to LU-2 devices. | D-26 |
| MDFYLIM | Specifies the number of concurrent system console MODIFY commands supported (OS/VS systems only). | D-26 |
| MENULU1 | Specifies an alternate soft menu for LU1 logons. | D-27 |
| MENUPROC | Specifies an alternate primary menu procedure name. | D-27 |
| MODDEL | Specifies the name of a load module or phase which is to be deleted from storage. | D-27 |
| MODLOAD | Specifies the name of a module to be loaded into the Management Services region. | D-27 |
| MODLUSER | Specifies the name of a UAMS user ID definition, present on the UAMS dataset, to be used as a model for dynamic user generation. | D-27 |
| MSRETAIN | Shows how many unused loaded maps are to be retained in storage, or gives an upper limit on the amount of storage they can take up. | D-28 |
| M2RETRY | Specifies that if Management Services attempts to start a session with a 3270 display terminal and the BIND fails, then it retries the connection designating the terminal as a model 2 device. | D-28 |
| NCLCSAVE | Specifies the default mode for the NCL &CONTROL SAVE/NOSAVE function. | D-28 |
| NCLEX01 | Specifies the NCL general authorization exit. | D-28 |
| NCLGBTRC | Specifies a single global variable name, or a generic global variable prefix, that is to be traced as changes occur. | D-28 |

*Table D-1. SYSPARMS Summary Table*

| SYSPARM | Description | See page |
|---|---|---|
| NCLGLBL | Specifies the NCL global variable prefix. | D-29 |
| NCLMAXK | Specifies the maximum amount of virtual storage that can be allocated to NCL processing at any one time. | D-29 |
| NCLOGTRM | Specifies whether NCL writes log message N03906 on completion of each NCL procedure. | D-29 |
| NCLPEND | Specifies whether an NCL process, executed while another serial NCL process is operating in the same OCS window, is to be queued pending completion of the existing procedure, or rejected because the earlier process is active. | D-30 |
| NCLPRSHR | Specifies the action performed when a request for concurrent or consecutive execution of the same NCL procedure is made. | D-30 |
| NCLTRLFF | Specifies how many X'FF' field separators NCL places at the end of a record written to a UDB. | D-30 |
| NCLTRMAX | Specifies the number of NCL trace messages that are generated when an NCL procedure is invoked. | D-30 |
| NCLUMAX | Specifies the number of NCL processes that can be executed for one user across all NCL processing environments. | D-31 |
| NCLXUSER | Specifies whether the INTQ, GO, FLUSH, and END commands can be used to send data to an NCL procedure operating only within the issuer's processing environment, or to any NCL procedure. | D-31 |
| NDBDIOL | Sets the default logical I/O limit to be placed on an &NDBSCAN statement. | D-31 |
| NDBDRCL | Sets the default limit for the number of records that can satisfy an &NDBSCAN statement. | D-31 |
| NDBDSTL | Sets the default limit for the amount of working storage that can be used while processing an &NDBSCAN statement. | D-31 |
| NDBDTML | Sets the default limit that can elapse when processing an &NDBSCAN statement. | D-32 |
| NDBSCANO | Enables or disables the scan optimizer. | D-32 |
| NDBMIOL | Sets the maximum logical I/O limit allowed by an &NDBSCAN statement. | D-32 |
| NDBMRCL | Sets the maximum limit for the number of records that can satisfy an &NDBSCAN statement. | D-33 |

*Table D-1. SYSPARMS Summary Table*

| SYSPARM | Description | See page |
|---|---|---|
| NDBMSTL | Sets the maximum limit for the amount of storage that can be used while processing an &NDBSCAN statement. | D-33 |
| NDBMTML | Sets the default limit that can elapse when processing an &NDBSCAN statement. | D-33 |
| NDBSUBMN | Sets the minimum number of subthreads that will stay active, for any NDB, awaiting database requests that can run asynchronously. | D-34 |
| NDBSUBMX | Sets the maximum number of subthreads allowed. | D-34 |
| NDBLOGSX | 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. | D-34 |
| NDBPHONX | Registers the name of the NCL phonetic exit program. | D-34 |
| NDBOPENX | Controls whether the nominated NCLEX01 is called for &NDBOPEN. | D-34 |
| NMIQLIM | Specifies the internal queuing limits for traffic. | D-35 |
| NONSWAP | Specifies whether this system is to run non-swappable or swappable (OS/VS systems only). | D-35 |
| NRDLIM | Specifies the number of non-roll delete messages that the system queues before deleting the oldest messages. | D-35 |
| NSPRSPTO | Sets the interval between retries when trying to connect to a NetSpy system. | D-35 |
| NSPRTINT | Sets the default timeout for solicited responses to commands sent to a NetSpy system. | D-35 |
| OCSHLITE | Specifies the type of highlighting to be used for messages appearing in OCS windows. | D-36 |
| OCSTIME | Specifies whether the time appears at the end of the title line of an OCS window. | D-36 |
| OMLUMAX | Specifies the maximum number of concurrent OML processes that can execute for one user across all OML processing environments. | D-36 |
| OSHOST | Specifies the default APPC linkname used to service object services requests. | D-36 |
| PANLBFSZ | Specifies the maximum outbound datastream size that can be generated for any terminal attached to Management Services. | D-37 |

*Table D-1. SYSPARMS Summary Table*

| SYSPARM | Description | See page |
|---|---|---|
| PANLBUFF | Specifies the maximum number of pages of virtual storage that can be used for concurrent terminal output operations. | D-37 |
| PANLPDD | Specifies the default preparse option to be used by panel services during preparse processing. | D-37 |
| PPOACBNM | Allows you to dynamically change the VTAM PPO ACB used by Management Services. | D-37 |
| PPOCOLOR | Specifies the color to be used when displaying unsolicited VTAM (PPO) messages on OCS consoles. | D-37 |
| PPOHLITE | Specifies the type of highlighting used when displaying unsolicited VTAM messages on OCS consoles. | D-37 |
| PPOPROC | Specifies the name of an NCL system procedure that is to be used to intercept and process unsolicited VTAM messages. | D-38 |
| PPOSOCMD | Specifies whether VTAM commands received from PPO as copies of commands entered from a system console are to be logged, passed to PPOPROC, or ignored. | D-38 |
| 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 to be logged. | D-38 |
| PPOUSMSG | Specifies how unsolicited PPO messages are to be written to the Management Services log, regardless of DEFMSG options. | D-39 |
| PRELOAD | Specifies the name of an NCL procedure that is to be loaded before execution. | D-39 |
| PRIACBNM | Allows you to dynamically change the name of the VTAM APPL definition used by Management Services as its primary ACB. | D-40 |
| PWEXPIRE | Specifies the number of days after which users are forced to change their password. | D-40 |
| PWMAX | Specifies the maximum acceptable length for passwords. | D-40 |
| PWMIN | Specifies the minimum acceptable length for passwords. | D-40 |
| PWRETRY | Specifies the number of times an incorrect password is accepted before a logon attempt is denied. | D-40 |
| RECONN | Specifies whether disconnected users can reconnect from any terminal or only the terminal from which they were initially signed on. | D-40 |

*Table D-1. SYSPARMS Summary Table*

| SYSPARM | Description | See page |
|---|---|---|
| ROFPAREN | Specifies the installation default ROUTE and SIGNON command parenthesis option. | D-40 |
| ROFPREFX | Specifies the installation default ROUTE and SIGNON command prefix option. | D-41 |
| ROUTCDE | Specifies the operating system routing codes to be used for unsolicited messages sent to the system console. | D-41 |
| RTA480 | Prevents Repeat To Address 3270 datastream commands, with a multiplier of more than 480, being generated. | D-41 |
| SESSMSG | Specifies whether trace message N35007 is issued each time a session to a remote system is opened or fails. | D-41 |
| SMFID | Specifies the SMF record identifier to be used in the generation of SMF messages. | D-42 |
| SSID | Controls SSI connections (OS/VS systems only). | D-42 |
| SSIRTINT | Specifies the retry interval for SSI connection attempts (OS/VS systems only). | D-42 |
| 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. | D-42 |
| 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. | D-42 |
| SYSCONMU | Specifies the default user ID for a master console user when it is not signed on. | D-43 |
| SYSCONNM | Specifies the LU name that is assigned to system console environments (MVS/ESA systems only). | D-43 |
| SYSCONSO | Specifies whether or not the console user ID is allowed to default, and whether signon is required or not (MVS/ESA systems only). | D-43 |
| SYSCONUI | Specifies the default system console signon name. | D-44 |
| SYSCONXU | Specifies whether or not external console user IDs are to be used when signing on consoles (MVS/ESA systems only). | D-44 |
| SYSLOG | Specifies whether none, all, or unsolicited VTAM messages written to the Management Services log, are to be written to the system log (OS/VS systems only). | D-44 |

*Table D-1. SYSPARMS Summary Table*

| SYSPARM | Description | See page |
|---|---|---|
| SYSLOGFM | Specifies the format of the SYSLOG lines (OS/VS systems only). | D-46 |
| SYSUDB | Specifies whether NCL procedures are allowed to access the Management Services VSAM datasets and treat them as UDBs. | D-46 |
| TCMDMAX | Specifies the maximum number of concurrently-active timer commands the system allows. | D-46 |
| TELNET | Activates or deactivates the Telnet server. | D-46 |
| TNDSREG | Specifies whether the Telnet server will register new connections with the Data Space Manager. | D-46 |
| TNGPREFX | Specifies the prefix to be used for generic Telnet sessions. | D-47 |
| TNSPREFX | Specifies the prefix to be used for specific Telnet sessions. | D-47 |
| TOACT1 | Specifies the action to be taken for a user when their first terminal timeout interval expires. | D-48 |
| TOACT2 | Specifies the action to be taken for a user when their second terminal timeout interval expires. | D-48 |
| TOALARM | Specifies the number of times the alarm rings if ALARM is the action specified in TOACT1 or TOACT2. | D-48 |
| TODISC | Specifies timeout disconnect values. | D-48 |
| TOLOCK | Specifies whether timeout actions (specified in TOACT1 or TOACT2) apply to a locked terminal. | D-48 |
| TOSTART | Starts or stops terminal inactivity timeout processing. | D-48 |
| TOTIME1 | Specifies the interval used at the first level timeout period for inactive terminals. | D-49 |
| TOTIME2 | Specifies the interval used at the second level timeout period for inactive terminals. | D-49 |
| TPEND | Specifies the action required by Management Services if its TPEND exit is driven. | D-49 |
| TRACEOPT | Specifies the trace options to be applied when tracing data streams sent to or from a terminal. | D-50 |
| UNLOAD | Specifies the name of an NCL procedure, or generic procedure name, that is to be unloaded. | D-50 |

*Table D-1. SYSPARMS Summary Table*

| SYSPARM | Description | See page |
|---|---|---|
| UNSPVPSZ | Specifies whether Management Services is to enforce the unspecified viewport size BIND option when establishing session with display terminals which support the Read-Partition-Query command. | D-51 |
| USERPW | Specifies whether the NCL system variable &USERPW is available for use in MAI logon data. | D-51 |
| VDISPLAY | Specifies how the VTAM display command (D) is processed for users with command network partitioning. | D-51 |
| VSAMCAT | Specifies the DLBL name for the VSAM catalog (VM systems only). | D-52 |
| VSAMMODE | Specifies the default mode character used for dataset allocation requests (VM systems only). | D-52 |
| VTAMAUTH | Specifies whether Management Services is to run using VTAM authorized path facilities. | D-52 |
| VTAMID | Specifies the system procedure name used for starting VTAM. | D-52 |
| VTAMCDRM | Specifies the name of the VTAMCDRM. | D-52 |
| VTAMNTFY | Specifies whether Management Services waits for confirmation of acceptance of a CLSDST/PASS function when passing an EASINET terminal to a target application (not applicable to VOS3 systems). | D-52 |
| XABELOW | Specifies whether Management Services is to allocate buffer storage below the 16MB line in XA systems if all storage in the extended private area has been used. | D-53 |
| XNFRETRY | Specifies whether Management Services is to retry the XNF ENTRY process if the first attempt fails (VOS3 XNF systems only). | D-53 |
| XNFSCRSZ | Specifies the default screen size for connected terminals (VOS3 XNF systems only). | D-53 |
| XNFUCENM | Allows you to dynamically change the name of the XNF UCE used by Management Services for XNF connectivity (VOS3 XNF systems only). | D-53 |

## Generic SYSPARMS Operand Descriptions

**ABENDCD=*code* | 4095 (OS/VS or VM systems)**

Specifies the user abend code that Management Services uses when it abends. If Management Services abends due to some internal error condition, it issues a user abend with this code. A console message provides further information on the reason for the abend. If this operand is not specified, it defaults to 4095.

**ACBRETRY=YES | NO**

Indicates whether Management Services 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. The default is YES.

**APPLSTAT=*name***

Specifies the name of a VTAM APPL definition for which automatic status checking is performed. The status of this application is automatically monitored at a frequency determined by the APPLSTIV operand. The application is always in one of six possible status conditions which correspond to VTAM's view of the application. By default these status conditions are:

- ACTIVE
- INACTIVE
- NOLOGONS
- UNKNOWN
- UNAVAIL
- NOTAPPL

The status of the application is made available to NCL procedures in an NCL global variable named:

`&&000applname`

where the *applname* suffix is the name of the APPL definition specified on the APPLSTAT operand. If using the SYSPARMS NCLGLBL operand to change the global variable identifier to other than its default of GLBL, set the replacement value before using the first APPLSTAT operand.

You can change the status text using the SYSPARMS APPTXT*n* operand (where *n* corresponds to the status conditions listed above) to make the description of the status more meaningful to your installation.

Each APPLSTAT specifies one application. One APPLSTAT must be coded for each application to be monitored. See the *SOLVE:Access Implementation and Administration Guide* for further information.

**APPLSTIV=<u>60</u> | *number***

Specifies the time interval at which automatic status checking of applications specified in the APPLSTAT operand is performed. The default time interval is 60 seconds, however, any value of 30 to 3600 seconds is valid. See the *SOLVE:Access Implementation and Administration Guide* for further information.

**APPTXT1=<u>ACTIVE</u> | *text***

Assigns a word from 1 to 24 characters long which describes the status of an application monitored by APPLSTAT processing when the application status corresponds to a VTAM status of ACTIVE.

When APPLSTAT monitoring determines that an application has a VTAM status of ACTIVE, it primes the global variable `&&000applname` with the value of APPTXT1. This allows a meaningful description of the application status to be given to network users, in English or any other language. If the application status is to be displayed on DBCS terminals, APPTXT1 can be set to contain a DBCS value. In this case the value appears as '.....' on non-DBCS devices.

**APPTXT2=<u>INACTIVE</u> | *text***

Assigns a word from 1 to 24 characters long which describes the status of an application monitored by APPLSTAT processing when the application status corresponds to a VTAM status of INACTIVE.

When APPLSTAT monitoring determines that an application has a VTAM status of INACTIVE, it primes the global variable `&&000applname` with the value of APPTXT2. This allows a meaningful description of the application status to be given to network users, in English or any other language. If the application status is to be displayed on DBCS terminals APPTXT2 can be set to contain a DBCS value. In this case the value appears as '.....' on non-DBCS devices.

**APPTXT3=<u>NOLOGONS</u> | *text***

Assigns a word from 1 to 24 characters long which describes the status of an application monitored by APPLSTAT processing when the application status corresponds to a VTAM status of NOLOGONS (meaning that the application is not accepting logon requests).

When APPLSTAT monitoring determines that an application has a VTAM status of NOLOGONS it primes the global variable `&&000applname` with the value of APPTXT3. This allows a meaningful description of the application status to be given to network users, in English or any other language. If the application status is to be displayed on DBCS terminals APPTXT3 can be set to contain a DBCS value. In this case the value appears as '.....' on non-DBCS devices.

**APPTXT4=<u>UNKNOWN</u> | *text***

Assigns a word from 1 to 24 characters long which describes the status of an application monitored by APPLSTAT processing when the application status corresponds to a VTAM status of UNKNOWN (meaning that the nominated *applname* is not defined to VTAM, or that the status has not yet been determined).

When APPLSTAT monitoring determines that an application has a VTAM status of UNKNOWN it primes the global variable `&&000applname` with the value of APPTXT4. This allows a meaningful description of the application status to be given to network users, in English or any other language. If the application status is to be displayed on DBCS terminals APPTXT4 can be set to contain a DBCS value. In this case the value appears as '.....' on non-DBCS devices.

**APPTXT5=<u>UNAVAIL</u> | *text***

Assigns a single word from 1 to 24 characters long which describes the status of an application monitored by APPLSTAT processing when the application status corresponds to a VTAM status of UNAVAIL (meaning that the nominated application is defined to VTAM but cannot be logged onto).

When APPLSTAT monitoring determines that an application has a VTAM status of UNAVAIL it primes the global variable `&&000applname` with the value of APPTXT5. This allows a meaningful description of the application status to be given to network users, in English or any other language. If the application status is to be displayed on DBCS terminals, APPTXT5 can be set to contain a DBCS value. In this case the value appears as '.....' on non-DBCS devices.

**APPTXT6=<u>NOTAPPL</u> | *text***

Assigns a single word from 1 to 24 characters long which is used to describe the status of an application monitored by APPLSTAT processing when the application status corresponds to a VTAM status of NOTAPPL (meaning that the nominated resource is not defined to VTAM as an application but as some other resource type).

When APPLSTAT monitoring determines that an application has a VTAM status of NOTAPPL, it primes the global variable `&&000applname` with the value of APPTXT6. This allows a meaningful description of the application status to be given to network users, in English or any other language. If the application status is to be displayed on DBCS terminals, APPTXT6 can be set to contain a DBCS value. In this case the value appears as '.....' on non-DBCS devices.

**AUTOEXEC=<u>YES</u> | NO**

If an unrecognized command string is entered from an OCS terminal, this operand specifies whether Management Services 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), Management Services 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.

**CALLSHR0=YES | NO (OS/VS and VM Systems only)**

Specifies whether subpool zero is to be shared between &CALL subtasks and the Management Services main task.

When &CALL statements are executed, Management Services attaches a subtask to execute the target load module. By default, the subtask does not share subpool zero with the main task. The default of NO should be satisfactory in most cases. However, if multiple called modules open and process the same VSAM dataset, VSAM can suffer abends unless subpool zero is shared.

**CDELAY=15 | *number***

Specifies the time Management Services waits before sending output to an OCS terminal when a user is entering input on the keyboard. The default is 15 seconds, however, any value of 0 to 300 seconds is valid.

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.

Management Services breaks this contention 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.

**CMDAUTH=( *command,cmdauth* [*,opauth*] )**

Used to change authority levels for any command or command operand.

Management Services (MS) commands have authority levels to limit their use to user IDs with a command authority level that is equal to or higher than that assigned to the command (*cmdauth*). Additionally, some command operands can be further restricted by specifying a higher authority level (*opauth*) than the command itself. Default authority level values are listed in Appendix H, *Command Authority Levels*.

The command name specified can be in its abbreviated form, for example:

```
SYSPARMS CMDAUTH=(REPLY,5) CMDAUTH=(NPTAB,0,8)
```

Valid authority values range from 0 to 255. CMDAUTH itself requires a command authority of 9 unless specifically changed during initialization. Specifying an *opauth* value for a command without special operand authority possibilities, results in an error.

CMDAUTH can also be used to disable a command completely by allocating it an authority higher than any defined for a user ID.

**Warning**

Commands with a default authority of zero (0) should not be altered as this can cause unpredictable results in the supplied applications.

**CMDREPL=*commandname***

Specifies an NCL procedure to run in the place of a specified command. The command and the NCL procedure must have the same name. This lets you replace commands with logic of your own.

If this operand is specified, the system searches the commands dataset for an NCL procedure with the name *commandname*. If a procedure is found it is preloaded and then automatically executed instead of the command. The NCL procedure itself can issue the native command (in either the primary or dependent environment), but in all other cases the NCL procedure is started instead of the command. If no procedure is found, the command is executed as normal.

The CMDREPL operand can be entered at any time during system execution allowing dynamic control of command replacement and interception.

To reverse a CMDREPL command, use the SYSPARMS UNLOAD operand to delete the appropriate NCL procedure from storage.

**Note**

If you are using CMDREPL to intercept the processing of certain commands, ensure that the SYSPARMS operand itself is suitably protected against unauthorized use.

You cannot replace the EXEC, FSPROC, or START commands.

**CMDSTACK=<u>6</u> | *n***

Specifies the size of the OCS command stack. Commands retained in the command stack can be recalled to the command line with the CS+ and CS- commands (assigned by default to F10 and F11). The value of *n* can be from 1 to 9; the default is 6.

**CONMSG=YES | NO**

Specifies whether Management Services 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.

**CONUIDPW=YES | NO**

Specifies whether Management Services is to accept user data when a user logs on to the system. If NO is specified, Management Services ignores all user data. In this case, the required user ID and password must be entered from the standard Management Services logon panel.

If YES is specified, three items of user data are accepted by Management Services: a user ID, password and initial menu selection. If this data is present (and the user ID and password are correct), the initial logon panel is bypassed and the requested menu displayed. The format of the data is defined in the *SOLVE:Access Implementation and Administration Guide*.

The YES option should be used only if the password that is provided at logon time can be entered in a secure (non-display) field on the terminal. The EASINET feature allows secure logon panels to be displayed and this option is designed for use with terminals under EASINET control.

In addition, user data consisting of the string NOEASINET causes a terminal to bypass the EASINET facility; if the first word of the user data is the string EASINET, the following data is tokenized and made available to the EASINET procedure in variables.

Further information on the use of logon user data by Management Services can be found in the *SOLVE:Access Implementation and Administration Guide*.

**DALDEFER=NO | YES (for OS/VS systems only, but not VOS3)**

Specifies whether deferred mounting is to be requested when allocating datasets. The default is NO.

**DALRACF=NO | YES (for OS/VS systems only, but not VOS3)**

Specifies whether automatic RACF resource protection is requested when dynamically allocating new datasets. The default is NO.

**DALRLSE=NO | YES (for OS/VS systems only)**

Specifies whether datasets created by dynamic allocation are defined with the RLSE option to free unused secondary extents. The default is NO.

**DBCS=<u>NO</u> | YES | IBM | FUJITSU**

Determines whether the system is to support devices that are capable of operating with a Double Byte Character Set (DBCS).

If DBCS devices are to be used to connect to Management Services, this operand must be coded. However, there are differences in DBCS implementations between IBM and Fujitsu devices. See the section titled *Customizing Management Services for Use with DBCS* in Chapter 2, *Customizing Management Services*, before coding this operand.

**DESC=(*n,n,n*) | <u>NONE</u> (for OS/VS systems only)**

Defines the operating system description codes used for messages sent to the system console, including messages associated with the *pppp*OPER user ID. The range is 1 to 16 and no default applies. See the appropriate operating system manual for an explanation of the impact of specifying certain codes. Specify NONE to cancel any previously set description codes.

**DYNLMAX=<u>10</u> | *number***

Specifies the maximum number of dynamic INMC links that can be active at the same time. The default is 10. Enter 0 to prevent any dynamic links from being established and to disable the dynamic facility.

**EASINET=<u>NO</u> | *procname***

Specifies whether Management Services is to invoke the EASINET feature. This operand only works if the system is configured with the EASINET feature.

If you require EASINET, specify the name of the NCL procedure that is to manage EASINET terminals. See the *SOLVE:Access Implementation and Administration Guide* for information on developing this procedure.

If NO is specified, EASINET cannot be activated after initialization. Similarly, if a procedure name is specified (and a new procedure can be specified at any time), NO cannot be specified to disable EASINET after initialization.

**EDITCAPS=<u>OFF</u> | 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.

**Note**

If running with the DBCS=YES option, the EDITCAPS function is ignored.

**EDITMAXK=32 | *number* | NONE**

Specifies the maximum number of kilobytes (Kb) of storage the system can allow for the Management Services editor at any time.

If you have multiple concurrent users of the Management Services editor, significant amounts of storage could be used. This parameter allows you to place a ceiling on the amount of storage used for this purpose.

If 0 is specified, no limit is imposed. If NONE is specified, editing functions are disabled. Other values are rounded up to the nearest 8Kb boundary. The *number* option is designed for use where one system is configured to allow updating of the PANELS dataset, while other systems sharing that dataset can only read it. The maximum value is 1000Kb.

**Note**

In MVS/XA systems all editor-related storage is held in the extended private area of the Management Services address space. Editor storage problems are unlikely in this environment, unless the size of the area is significantly restricted.

**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.

**EVCMDMIN=10 | *number***

Determines the minimum repeat frequency permitted for an EVERY command. This can be used to stop the system being flooded with timer commands inadvertently. The range is 0 to 300 seconds, the default is 10. If a value of 0 is specified, an EVERY command is executed immediately.

**GENRSRC=*xxxxxxx***

Defines the generic resource group to which this region is to belong. The region is enrolled as a member of this generic resource group during initialization.

This SYSPARM must be set in RMINIT. If no value is set, the region does not participate in a generic resource group.

**HELDMSG=(<u>200,999</u>) | (*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 is (200,999); the range is 10 to 999. The value of *xxx* cannot exceed the value of *yyy*.

**ID=*hostid***

Defines the 1- to 12-character system identifier. This name identifies the system, and is used in status messages associated with links to other systems. If this operand is omitted, the default name assigned is that of the VTAM APPL definition the system uses, as specified on the PRI JCL parameter (or the default, NM, if not specified). If the APPL name is not particularly meaningful, it is recommended you choose a system name that helps identify the system, particularly if multiple systems are operating within your network. Check the NMDID JCL parameter, as this provides a short domain ID used by many facilities.

**Note**

We recommend that you use only alphanumeric characters in the system identifier.

**INMCBFSZ=<u>4</u> | *number***

Specifies the INMC buffer size (in K) 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.

The default value for INMCBSZ is 4. 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.

**INMCEX01=*exitname* | <u>NONE</u>**

Defines the load module for the INMC Primary Security Exit to be invoked whenever any INMC link becomes active. The default is NONE, and *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.

**INMCEX02=*exitname* | NONE**

Defines the load module for the INMC Secondary Security Exit to be invoked whenever any INMC link becomes active. The default is NONE, and *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.

**IPAMHB={ NO | (*a*,*b*) }**

Controls the use of heartbeats for TCP/IP INMC and APPC links.

**Note**

Heartbeats are used only if both sides of a link are using Management Services Version 3.7 with fix Z41351, or later.

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.

**IPCHECK=REGISTER | VERIFY | NONE**

Controls the level of functionality required for associating IP addresses with user IDs.

IPCHECK=REGISTER results in user IDs being recorded for IP connections by signon and &SECCALL CHECK processing.

IPCHECK=VERIFY results in validation of a user's right to access the system during signon and &SECCALL CHECK processing. REGISTER is the default for all systems except those running NetMaster for TCP/IP Access Control, where VERIFY is the default.

If IPCHECK=VERIFY is set and a data space subsystem is defined and the data space is unavailable, then all logons fail, with a message indicating that the required data space is not active. If IPCHECK=VERIFY is set but no data space subsystem is defined to Management Services, then IPCHECK is treated as being NONE.

**IPPORT=*n* | NONE**

Specifies information to enable the sockets interface to IBM TCP/IP or TCPaccess where *n* identifies the IP port number (in the range 257 to 65535) that the shared port server is to use for your TCP/IP system to listen for connection requests. The default is 2636. This number is registered with the Internet Assigned Numbers Authority (IANA). Specify NONE to prevent this system from listening for incoming connections from other systems.

If the IPPORT value is set to 0, then no shared port is available, and the Telnet server is not available on this port.

**Note**

A TNCTL ADDPORT=SHARED command must also be issued to activate the TELNET server for this port.

**JRNLPROC=$NDJPROC | *procname***

Specifies the NCL procedure to be started when a journal swap occurs. The default procedure name is $NDJPROC.

**JRNLSWAP=YES | NO**

Specifies whether the NDB journal dataset 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.

**LANG=US | UK | *cc***

Defines the language code for the system and the default language code for users.

See the *Network Control Language Reference* manual for the system recognized values that can be used to replace the *cc* option on this operand. Other values can be specified; however, the default language code is US.

**LMSGWARN=10 | *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. The range is 1 to 999, the default is 10.

**LNKTRACE=YES | NO**

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.

**LOCKMAX=<u>128</u> | *number***

Specifies the maximum number of &LOCK requests that can be active in the system concurrently. &LOCK lets NCL processes serialize access to resources within the system. Values of 0 to 32767 can be specified. The default is 128.

**LOCKPROC=<u>$NMLOCK</u> | *procname***

Specifies the LOCK command procedure. The default is the Management Services supplied procedure $NMLOCK. This procedure is invoked when a Management Services user enters the LOCK command to lock the terminal.

**LOGPAGE=<u>60</u> | *number***

Defines the number of lines per page of the activity log. This operand must be executed near the start of the RMINIT initialization procedure to become immediately effective. Values of 30 to 250 can be specified. The default is 60.

**LOGPROC=*procname* | FLUSH**

Specifies the NCL system procedure in the Management Services procedures dataset that is used to intercept and process all messages issued to the activity log. All messages are passed to this procedure. It can then delete messages or change their format. All NCL facilities are available for use in this procedure. Any valid NCL procedure name can be specified—no default applies.

This command loads the LOGPROC procedure and commences processing. Specifying LOGPROC=FLUSH terminates the current LOGPROC.

**LOGWRAP=YES | <u>NO</u> (for OS/VS systems only)**

Specifies whether the log datasets can be reused. If NO is specified, a log dataset cannot be reused following a LOGSWAP command. If YES is specified, the system reuses log datasets if required. See Chapter 12, *Implementing Activity Logs,*for further details.

**LU1BKSP=YES | <u>NO</u>**

Specifies what to do with the backspace character (X’16’) when reading data from an LU1 device. YES indicates that the backspace character and its preceding character are received from the device. NO (the default) indicates that the backspace character and its preceding character are suppressed.

**MAIACBOR=*nn***

An MAI ACB name is constructed using a prefix and a numeric suffix. When an open fails, 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 open fails in an error situation.

**Note**

When an open fails in a non-error situation, the retry limit is reset to the value of this operand.

The range of values is 5 to 99, and the default is 5.

**MAIEX02=*exitname* | NO**

Specifies the name of an exit routine to take control whenever an MAI-OC session is started or ended. The exit routine also receives notification of the LU name chosen by MAI. This routine is given details of 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. See Appendix L, *MAI Installation Exit MAIEX02*, for further details. 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-OC 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. See Appendix L, *MAI Installation Exit MAIEX02*, for further details.

**MAIFILE={ *UDBname* }**

Specifies the UDB name of the MAI session definition file. A null value is treated the same as specifying VFS. This parameter is used by the MAI System Parameters option of System Support Services for MAI : Services, and MAI initialization when you have specified a non-VFS file for the MAI session definition file.

**MAIONL=<u>YES</u> | NO**

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.

**MAIOPREF=*prefix***

Specifies a 1- to 5-character string which is used as the prefix to an LU name generated by MAI-OC. For example:

```
SYSPARMS MAIOPREF=MAI02
SYSPARMS MAIOPREF=NMT
```

If the SYSPARMS MAIOPREF operand is not coded, MAI-OC assumes a default prefix of NMMAV.

**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)
```

**MAPDEL=*mapname***

Removes a loaded Mapping Services map from memory. If the map has active users when the command is issued, it is not unloaded until the active use count becomes zero. The map can be reloaded using the SYSPARMS MAPLOAD operand.

**MAPFILE=*filename***

Specifies the file used by Mapping Services to store and load maps. The initial setting is OSCNTL.

**MAPLOAD=*mapname***

Loads a Mapping Services map into memory. The command has no effect if the map is already loaded when the command is issued (that is, the map is not re-loaded). Note that maps are automatically loaded when they are referenced from an NCL procedure.

**MAPRESET=*mapname***

Queues a Mapping Services map for removal from memory and loads a fresh version of the map. All new attachments to the map from NCL procedures are made to the refreshed map. Attachments which exist when the command is issued remain attached to the old map. When all attachments to the old map have ended (that is, the active use count becomes zero), it is removed from memory. This command can be used to introduce an updated version of a map into the system.

**MAXPANEL=50 | *n***

Specifies the number of panels held in storage. For performance reasons, Management Services stores copies of the last *n* different full-screen panel skeletons that have been displayed (*n* being the number specified on the MAXPANEL operand). As the panels are displayed, the least-used ones are subjected to an ageing process which deletes them from storage in favour of the more heavily used panels. Increasing this number lessens the frequency with which fresh copies of panels have to be retrieved from the PANELS database, but also increases the storage requirements of the system. This figure might need to be tuned if multi-panel EASINET applications are in use and excessive refresh rates are suspected.

If 0 is specified, no limit is to be placed on the number of panels held in storage. If panel definitions are modified on the panels dataset using the editor, copies of those panels held in storage at the time are automatically invalidated.

**Note**

If a panels dataset is shared between two or more systems, the invalidation process occurs only on the system on which the panel was modified.

**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.

**MAXRUT2=32768 | *number***

Specifies the maximum request unit size used when sending chained output to LU-2 devices. If necessary, this value overrides session BIND parameters. The default is the maximum possible RU size.

**MDFYLIM=5 | *number* (for OS/VS systems only)**

Specifies the number of concurrent system console MODIFY commands supported. If 0 is specified, the system console interface is deactivated. The range is 0 to 255, the default is 5.

**MENULU1=$NMPMLU1 | *procname***

Specifies an alternate menu for LU1 logons. The default is the Management Services supplied procedure, $NMPMLU1.

**MENUPROC=$NMPMENU | *procname***

Specifies an alternate primary menu procedure name. The default is the Management Services supplied primary menu procedure, $NMPMENU.

**MODDEL=*modulename***

Specifies the name of a load module which is to be deleted from storage. This operand can be used only for modules loaded previously using the MODLOAD operand.

**MODLOAD=*modulename***

Specifies the name of a module to be loaded into the Management Services region. The MODLOAD operand lets you load the module in advance so that it is immediately available. At the completion of processing the module is deleted. If a module is invoked frequently, loading can cause delays. When the &CALL statement is used to invoke an external program, this must be loaded first before processing can begin.

**Note**

This option is effective ONLY if the target load module is link edited with RENT or REUS attributes. The operating system loads its own copy of the target module when &CALL is issued if the module does not have one of these attributes.

Once loaded with MODLOAD, a module remains in the region until the system is terminated or until a SYSPARMS MODDEL command is issued. MODLOAD can be used to pre-load many user-written modules, such as the various user exits, as long as they are reentrant or reusable.

**MODLUSER=*userid* | NONE**

Specifies the name of a UAMS user ID definition, present on the UAMS dataset, 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 allow it to default.
- If this operand is coded and a user attempts to log on to Management Services 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.

**MSRETAIN=*number***

Shows how many unused loaded maps are to be retained in storage, or gives an upper limit on the amount of storage they can take up. When the limit is reached, some unused maps are unloaded, until the storage is back within the limits.

**M2RETRY=YES | NO**

Specifies that if Management Services attempts to start a session with a 3270 display terminal and the BIND fails, then it retries the connection by designating the terminal as a Model 2 device. BIND failure indicates the terminal does not support the designated screen size, probably because the LOGMODE table entry for the terminal definition is wrong.

**NCLCSAVE=YES | NO**

Specifies the default mode for the NCL &CONTROL SAVE/NOSAVE function. The default is YES, which means that &CONTROL environments are restored on return from a nested NCL procedure.

**NCLEX01=*exitname* | NO**

Specifies the name of the NCL authorization exit. This exit provides security for NCL procedures and &SMFWRITE verbs. Full details of NCLEX01 are provided in Chapter 4, *Implementing Additional Security*.

If NCLEX01 is frequently invoked, use the MODLOAD operand to load a re-entrant version of the exit into the Management Services 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.

**NCLGLBL=<u>GLBL</u> | *cccc***

Specifies the NCL global variable prefix. Global variables are variables that are common throughout NCL procedures; they can be set in one procedure and tested in another.

The identifier prefix is the string used to identify variables which are to be regarded as global and all variables that start with this prefix are given global status. The string must be from 1 to 4 characters long and can commence with any non-numeric character that is valid when naming a variable (that is, A to Z, @, #, and $).

If application status monitoring is to be utilized, this operand must be used before the first APPLSTAT operand, as status monitoring creates global variables for referencing by NCL procedures. This operand is not valid after system initialization is complete.

**Note**

If you want to set your own global variable prefix make sure that you do not choose one that conflicts with other system values. For example, choosing a prefix such as: SYSPARMS NCLGLBL=Z causes any NCL variable starting with &Z to be treated as a global variable, conflicting with the system variables that start with &Z. It is not recommended to use a global variable prefix that starts with either &$ or &#, as this would conflict with distributed NCL procedures.

**NCLMAXK=<u>0</u> | *number***

Specifies the maximum amount of virtual storage in kilobytes, that can be allocated to NCL internal processing at any time. Values of 100 to 9999Kb are valid. If 0 is specified, no limit is placed on NCL storage use. Note that this does not restrict the amount of storage used for NCL variables.

**Note**

Preloaded NCL procedures use the storage constrained by this operand. If you use many preloaded procedures you might need to review this parameter if non-preloaded procedures fail because of storage shortages.

**NCLOGTRM=YES | <u>NO</u>**

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. The default is NO.

**NCLPEND=<u>YES</u> | NO**

Specifies whether an NCL process executed while another serial NCL process is operating in the same OCS window, is to be queued pending completion of the existing procedure (YES), or rejected because the earlier process is active (NO). This operand only applies to processes invoked by the serial execution EXEC command—it does not apply to START commands which execute asynchronous processes in parallel with other processes in the same NCL processing environment.

**NCLPRSHR=*number*M | *number*K**

Specifies the amount of storage set aside for retained NCL procedures. The default is 2048K.

Specifying a number in the range from 1 to 999 indicates that a request stack be maintained, where *number* NCL procedures are retained in storage after the last user has completed processing.

Specify a number from 1 to 9999, with a suffix of K, to limit the request stack size by storage allocation. This specifies the storage in kilobytes that is allocated to the stack.

These options give frequently used procedures a high probability of being stacked between execution requests, to limit the amount of disk I/O activity required. The retention stack is maintained, with the procedures least likely to be used being deleted first.

**Note**

For compatibility with earlier releases, which allowed a number of NCL procedures to be specified, NCLPRSHR=*number* is supported.

If a number of procedures is specified, that number is multiplied by 32K, for example, NCLPRSHR=30 is equivalent to 960K.

**NCLTRLFF=<u>ONE</u> | MULT**

Specifies how many X‘FF’ field separators NCL places at the end of a record written to a UDB. The default is ONE. 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.

**NCLTRMAX=<u>100</u> | *number***

Specifies the maximum number of NCL trace messages that are generated in any one invocation of an NCL procedure. The default is 100. Values of 0 to 9999 are valid. If 0 is specified, all tracing is inhibited.

**NCLUMAX=<u>128</u> | *number***

Specifies the maximum number of concurrent NCL processes that can be executed for one user across all NCL processing environments. The default is 128. Valid values are 1 to 32767.

**NCLXUSER=<u>YES</u> | NO**

Specifies whether the INTQ, GO, FLUSH, and END commands can be used to send data to an NCL procedure operating only within the issuer's processing environment (NO) or to any NCL procedure (YES).

**NDBDIOL=*n***

Sets the default logical I/O limit to be placed on an &NDBSCAN statement that does not have the IOLIMIT= parameter specified, or has it specified as 0 or null. The system default value is 2000.

The value specified can range from 0 to 1,500,000.

A logical I/O is a single request to VSAM. Do not be afraid to use high values, as, typically, it can take 10 logical I/Os to generate one physical I/O.

A scan exceeding the I/O limit is terminated with a warning response.

**Note**

Regardless of how the scan gets an I/O limit, it is always rounded down to the NDBMIOL value if it exceeds it.

**NDBDRCL=*n***

Sets the default limit for the number of records that can satisfy an &NDBSCAN statement that does not have RECORDLIMIT= specified, or has it specified as 0 or null.

The value specified can range from 1,000 to 1,000,000

If the number of records passing a given scan meets or exceeds this limit the scan is terminated with a warning response.

**Note**

Regardless of how the scan gets a record limit, it is always rounded down to the NDBMRCL value if it exceeds it.

**NDBDSTL=*n***

Sets the default limit (in Kilobytes) for the amount of working storage that can be used while processing an &NDBSCAN statement that does not have STGLIMIT= specified, or has it specified as 0 or null.

Scan working storage is used to hold intermediate result lists, bitmaps, and keys for sorting.

The value specified can range from 20 to 28,672.

If the amount of working storage used exceeds this value, the scan is terminated with a warning response.

**Note**

Regardless of how the scan gets a storage limit, it is always rounded down to the NDBMSTL value if it exceeds it.

**NDBDTML=*n***

Sets the default limit (in seconds) that can elapse when processing an &NDBSCAN statement that does not have TIMELIMIT= specified, or has it specified as 0 or null.

The value specified can range from 20 to 3,600

If the elapsed time exceeds this value, the scan is terminated with a warning response.

**Note**

Regardless of how the scan gets a time limit, it is always rounded down to the NDBMTML value if it exceeds it.

**NDBSCANO= {YES | NO}**

Enables (YES) or disables (NO) the scan optimizer.

**Note**

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 via the OPTIMIZE operand of the NDB START command.

**NDBMIOL=*n***

Sets the maximum logical I/O limit allowed by an &NDBSCAN statement. This value overrides any value provided, either on the statement, or by default from NDBDIOL, if the NDBMIOL value is lower.

A logical I/O is a single request to VSAM. Do not be afraid to use high values, as, typically, it can take 10 logical I/Os to generate one physical I/O.

The value specified can range from 10,000 to 1,500,000

A scan exceeding the I/O limit is terminated with a warning response.

**Note**

Regardless of how the scan gets an I/O limit, it is always rounded down to the NDBMIOL value if it exceeds it.

**NDBMRCL=*n***

Sets the maximum limit for the number of records that can satisfy an &NDBSCAN statement. This value overrides any value provided, either on the statement, or by default from NDBDRCL, if the NDBMRCL value is lower.

The value specified can range from 5,000 to 1,000,000

If the number of records passing a given scan meets or exceeds this limit the scan is terminated with a warning response.

**Note**

Regardless of how the scan gets a record limit, it is always rounded down to the NDBMRCL value if it exceeds it.

**NDBMSTL=*n***

Sets the maximum limit (in Kilobytes) for the amount of storage that can be used while processing an &NDBSCAN statement. This value overrides any value provided, either on the statement, or by default from NDBDSTL, if the NDBMSTL value is lower.

The value specified can range from 50 to 28,672.

If the amount of working storage used exceeds this value, the scan will be terminated with a warning response.

**Note**

Regardless of how the scan gets a storage limit, it is always rounded down to the NDBMSTL value if it exceeds it.

**NDBMTML=*n***

Sets the default limit (in seconds) that can elapse when processing an &NDBSCAN statement. This value overrides any value provided, either on the statement, or by default from NDBDTML, if the NDBMTML value is lower.

The value specified can range from 1 to 3,600.

If the elapsed time exceeds this value, the scan is terminated with a warning response.

**Note**

Regardless of how the scan gets a time limit, it is always rounded down to the NDBMTML value if it exceeds it.

**NDBSUBMN=*n***

Sets the minimum number of subthreads that will stay active, for any NDB, awaiting database requests that can run asynchronously (in ASM 3.0 &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.

The system default is 3. The value can range from 0 to 20.

**NDBSUBMX=*n***

Sets the maximum number of subthreads allowed. See NDBSUBMN above.

The system default is 5. The value can range from 1 to 20.

**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.

The system default is 40. The value can range from 10 to 200.

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. See the chapter, *Implementing User Programs*, for further details. The journal automatically extends if it is under-allocated.

**NDBPHONX=*name***

Registers the name of the NCL phonetic exit program.

**NDBOPENX={NO| YES}**

Controls whether the nominated NCLEX01 is called for &NDBOPEN.

**NMIQLIM=<u>1000</u> | *number***

Sets internal queueing limits for the following traffic:

- ROF messages queued to an INMC link
- Inbound messages from LU-1 terminals queued for processing by &PROMPT NCL statements
- Data queued to an NCL process's internal command environment by INTQ commands, or resulting from commands issued using &INTCMD
- Queue length of messages queued to PPOPROC
- Used to calculate the default ISR QMAX. (See the ISR QMAX command description in the *Management Services Command Reference.)*
- SNAMS flow control—used to control the number of messages queued to NCL processes and instances of the MDS_SEND TP

NMIQLIM is initially set to 1000. Values of 10 to 32767 are valid.

**NONSWAP=YES | NO (for OS/VS systems only)**

Specifies whether this system is to run non-swappable (YES), or swappable (NO). This operand is valid only if Management Services is running authorized.

For MVS and MSP, Management Services 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 Management Services 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 MVS and MSP optional features which require non-swappable operation blocks attempt to revert to swappable operation.

**NRDLIM=<u>200</u> | *number***

Specifies the maximum number of non-roll delete (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. The default is 200 messages. Values of 10 to 10000 messages are valid.

**NSPRTINT=*interval***

Sets the interval between retries when trying to connect to a NetSpy system. It is an integer number of seconds. The valid range is 10 to 600. The default is 30.

**NSPRSPTO=*timeout***

Sets the default timeout (in seconds) for solicited responses to commands sent to a NetSpy system. The valid range is 5 to 120 (integers only). The default is 10.

**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**

Messages are presented without change.

**REVERSE**

Messages are presented in reverse video.

**BLINK**

Messages are to blink.

**USCORE**

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.

**OCSTIME=YES | NO**

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.

**OMLUMAX=*n***

Specifies the maximum number of concurrent OML processes that can execute for one user across all OML processing environments. Values of 1 to 32767 are valid. The default is 128.

**OSHOST=*appclink* | NULL**

Identifies the default APPC linkname to be used to service object services requests. Object services requests are brokered with the &OSCALL NCL statement. Object services directory services determine the target linkname for the object request. If no registration exists for the object name, the value assigned to this operand is used. Specifying NULL indicates that the server is started in the local system.

**PANLBFSZ=8 | *number***

Specifies the maximum outbound datastream size that can be generated for any terminal attached to Management Services. The default is 8 Kb. Values of 4 to 20 Kb are valid.

If your network contains terminals that might receive large datastreams (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.

**PANLBUFF=40 | *number***

Specifies the maximum number of pages of virtual storage to be used for concurrent terminal output operations. The default is 40. Values of 0 to 32767 are valid. 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 Management Services functions.

**PANLPPD=DYNAMIC | STATIC**

Specifies the default preparse option to be used by Panel Services during preparse processing. Preparse processing can be dynamic or static.

This can be overridden by specifying the PREPARSE operand on the #OPT statement when defining a panel. See the description of the #OPT panel control statement in the *Network Control Language User's Guide*.

**PPOACBNM=*applname***

Allows the name of the VTAM PPO ACB, used by Management Services, to be changed dynamically. An attempt to change the name is rejected if the PPO ACB is currently open. To prevent a rejection of this type, issue PPO STOP to close the current ACB, change the name of the PPO ACB, and then issue PPO START to reopen the interface.

**PPOCOLOR=*color***

Specifies the color to be used when displaying unsolicited VTAM (PPO) messages on OCS consoles. The color can be any one of the following:

```
BLUE | RED | PINK | GREEN | TURQUOISE | YELLOW | WHITE | NONE
```

The default is white. PPO messages are displayed in high intensity on non-color devices.

**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.

**PPOPROC=*procname* | FLUSH**
Specifies the name of an NCL system procedure that is to be used to intercept and process unsolicited VTAM messages. Use the DEFMSG command to specify the messages this procedure is to intercept.

This operand loads the PPOPROC procedure and commences processing. Any messages generated by the procedure are sent to Monitor terminals prefixed with P to identify their origin.

Specifying PPOPROC=FLUSH terminates the current PPOPROC and resumes standard message delivery.

See your *Network Control Language User's Guide* for additional information on the PPOPROC procedure.

**PPOSOCMD=LOG | PPOPROC | IGNORE**
Specifies whether VTAM commands, received from PPO as copies of commands entered from a system console, are to be logged (the default), passed to PPOPROC, or ignored.

VTAM relays copies of commands entered through system consoles if a MODIFY PPOLOG=YES VTAM command has been entered, or if PPOLOG=YES is specified in the VTAM initialization parameters. VTAM commands entered from local OCS consoles are treated in the same way as if they had been entered from a system console.

**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).

**PRELOAD=*procname* | (*procname,ddname*)**
Specifies the name of an NCL procedure that is to be loaded before execution to avoid the overheads associated with loading it at execution time. *ddname* is an optional name of the DD job control statement that specifies the dataset from which the procedure is to be loaded (OS/VS) or the File Type of the dataset (VM). If *ddname* is not specified, the procedure is loaded from the default NCL library.

**Note**

A preloaded procedure can be unloaded using the SYSPARMS UNLOAD operand. Compare this operand and the NCLPRSHR operand. Procedures that replace native commands via the SYSPARMS CMDREPL operand are automatically preloaded. The PROFILE NCLTEST=YES command lets individual OCS users bypass procedure preloading and retention, to ensure that while testing they always obtain the latest copy of the procedure from disk.

**PRIACBNM=*applname***

Allows you to dynamically change the name of the VTAM APPL definition used by Management Services as its primary ACB. This operand can only be issued if the primary ACB is closed.

**Note**

The Management Services primary ACB name is passed to the UAMS security exit as one of the parameters for the system initialization call. This occurs before system initialization procedures are executed and is therefore always the ACB name as coded on, or defaulted by, the PRI JCL parameter.

**PWEXPIRE=30 | *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. The default is 30. Values of 1 to 366 days are valid. Enter a value of 0 to disable automatic password expiry.

**PWMAX=8 | *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. The default is 8. Values of PWMIN to 8 are valid.

**PWMIN=3 | *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. The default is 3. Any value from 1 to PWMAX is valid.

**PWRETRY=2 | *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. The default is 2. Values of 1 to 10 are valid.

**RECONN=ANY | SAME**

Specifies whether disconnected users can reconnected from any terminal (ANY) or only the terminal from which they were initially signed on (SAME).

**ROFPAREN=YES | NO**

Specifies the installation default ROUTE and SIGNON command parenthesis option. If YES is specified, the ROF message prefix is enclosed in parentheses.

**ROFPREFX=<u>BEFORE</u> | AFTER | NO**

Specifies the installation default ROUTE and SIGNON command prefix option. The following options can be specified:

**BEFORE**

The message prefix is to be added to ROF messages from remote systems before the message is seen by the user's MSGPROC.

**AFTER**

The message prefix is to be added to ROF messages from remote systems after the message is seen by the user's MSGPROC.

**NO**

No message prefix is to be added to ROF messages from remote systems.

This operand also determines whether the ROF prefix appears in the Management Services hard copy log. If it is set to NO, the prefix does not appear in the log, otherwise it is present.

**ROUTCDE=(<u>1,8,11</u>) | (*n,n,n*) (for OS/VS systems only)**

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). See the appropriate operating system manual for an explanation of the impact of specifying certain codes. The default is 1,8,11. Valid values are 1 to 16.

**RTA480=YES | <u>NO</u>**

Specifies whether *repeat to address* 3270 datastream commands, with a multiplier of more than 480, are generated. Specify YES if your network contains display devices that cannot accept these commands. These devices are generally older, often bisync devices.

**SESSMSG=YES | <u>NO</u>**

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.

**SMFID=0 | *number* (for OS/VS systems only)**

Specifies the SMF record identifier to be used in the generation of SMF messages. If Management Services is to generate SMF records on behalf of any of its components, specify the SMF record identifier that is to be used. Management Services generates a single SMF record type identified by this number. The range of numbers is governed by SMF user record type restrictions and must be between 128 and 255 inclusive, according to your installation's requirements. Specify 0 to suppress all SMF record generation. The default is 0.

The distributed NTS exit can generate SMF type 39 records, and the distributed NEWS procedure $NWSMF can generate SMF type 37 records, regardless of the setting of this operand. They create system-level SMF record numbers.

The &SMFWRITE NCL statement can be used by authorized NCL procedures to write SMF records. These records are not subject to any specification of this operand.

**SSID={ NO | * | *name* } (for OS/VS systems only)**

Controls SSI connections. Specifying NO terminates any current or pending SSI connection. If there is no current or pending SSI connection, it does nothing.

Specifying * (which uses the first 4 characters of the jobname), or *name* (specific SSID) initiates a connection to the indicated SSI. If a connection currently exists, the request is rejected with an error message. If a pending connection exists, it is immediately retried with the new SSID.

**SSIRTINT={ 30 | *n* } (for OS/VS systems only)**

Specifies the retry interval for SSI connection attempts. The default is 30 seconds. Valid values are 10 to 600 seconds. Each retry writes a monitor message displaying the failure reason.

**STGWRN=0 | *number***

Specifies the number of kilobytes below the 16 megabyte 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. The default is 0, indicating that no warning limit applies. Values of 0 to 16,000 are valid.

**STGWRNXA=0 | *number***

Specifies the number of kilobytes above the 16 megabyte 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. The default is 0, indicating that no warning limit applies. Values of 0 to 1,000,000 are valid.

**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. The *name* must be a valid 1- to 8-character user ID.

The default value for SYSCONMU at initialization is *pppp*MSOP, where *pppp* is the system user prefix.

SYSCONMU can be set only during RMINIT. 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.

See the section, *Defining a System Console User ID*, in Chapter 3, *Implementing Security*, for detailed information on master console user ID requirements.

**SYSCONNM=<u>EXTMCS</u> | ALL (for MVS/ESA systems only)**

Specifies the LU name that is assigned to system console environments. The following options can be specified:

**EXTMCS**

Consoles with console IDs in the range 0 to 99 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**

The LU name is always the MVS-assigned console name. This means that console 0 is INTERNAL, the master is MASTER, and so on.

**SYSCONSO=<u>DEFAULT</u> | NO | REQUIRED (MVS/ESA systems)**

Specifies whether or not the console user ID is allowed to default, and whether or not signon is required. It is relevant to all operating system environments. The following options can be specified:

**DEFAULT**

The default SYSCONUI user ID name is used to signon if the system console user ID is not defined in the UAMS security system.

**NO**

Signs on all consoles using the SYSCONUI user ID name.

**REQUIRED**

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.

**SYSCONUI=*name***

Specifies the default system console signon name. The *name* must be a valid 1 to 8 character user ID.

**Note**

The user ID is not used unless it exists in the security system in use.

See the section, *Defining a System Console User ID*, in Chapter 3, *Implementing Security*, for detailed information on console user ID requirements.

**SYSCONXU=NO | YES (for MVS/ESA systems only)**

Specifies whether or not external console user IDs are to be used when signing on consoles. This operand is relevant only in MVS/ESA 4.1, or later, environments. The following options can be specified:

**NO**

The system-supplied user ID is not used.

**YES**

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.

**SYSLOG=NO | YES | PPO (for OS/VS systems only)**

Specifies whether none, all, or unsolicited VTAM messages written to the Management Services log, are also written to the system log. The following options can be specified:

**NO**

No Management Services messages are written to the system log.

**YES**

Copies everything written to the Management Services log to the system log.

**PPO**

All unsolicited VTAM messages received by Management Services are written to the system log as well as to the Management Services log.

**SYSLOGFM=MVS | MSP | XA | ESA (for OS/VS systems only)**

Specifies the format of the SYSLOG lines produced if SYSLOG=YES or SYSLOG=PPO. The following options can be specified:

**MVS** and **MSP**

Format a line with four zeroes for ROUTCDE, time, and user ID in the JOBID column.

**XA** and **ESA**

Format a line with eight zeroes for ROUTCDE, Management Services domain ID for SMFID, date, time, user ID, and eight zeroes for MPF flags.

**SYSUDB=YES | NO**

Specifies whether NCL procedures are allowed to access Management Services VSAM datasets and treat them as UDBs. The default is YES.

**Note**

If the Management Services datasets are accessed, care must be taken not to corrupt information used by Management Services for internal operation.

**TCMDMAX=9999 | *number***

Specifies the maximum number of concurrently-active timer commands the system allows. Specifying 0 inhibits the use of any timer commands. The default is 9999. Valid values are 0 to 9999.

**TELNET=NO | YES**

Activates or deactivates the Telnet server. If it is set to NO (the default), then the Telnet Server is not activated. In this case, the Shared Port Server (and servers on other nominated Telnet ports) simply close any incoming connections that are not valid Shared Port Server requests.

This SYSPARM can be changed at any time. However, if it is set to NO, then any existing Telnet sessions are not closed, but new ones cannot be started.

**TNDSREG=NO | YES**

Specifies whether the Telnet Server will register new connections with the Data Space Manager, for use by any NetMaster for TCP/IP regions on the same MVS image.

The default is NO, meaning that new TELNET connections are not registered. Setting YES causes the connection to be registered with the Data Space Manager. If SYSPARMS IPCHECK is set to REGISTER or VERIFY, then the connection is augmented with the user ID when a user signs on to the domain.

**TNGPREFX=<u>TLNG</u> | *pppp***

Specifies the prefix to use for generic Telnet sessions. These are sessions where no specific terminal name was (or could be) requested. This includes the following:

- All NVT sessions
- TN3270 sessions (unless the client supports the RFC 1646 extensions where the required terminal name is separated from the terminal type by an @ sign)
- TN3270E sessions where no specific name was requested

The default is TLNG. The value must be exactly four characters, in PDSNAME format (alpha or national first, and then alpha, national, or numeric for the rest).

The actual name is formed from this prefix and a number from 0001 to 9999, thus allowing up to 9999 concurrent generic sessions. Suffixes are recycled, wrapping after 9999 is used; however, existing sessions are not reused.

**TNSPREFX=<u>TLNS</u> | *pppp***

Specifies the prefix to use for specific Telnet sessions. These are sessions where a specific terminal name is requested. This can occur only for TN3270E sessions and TN3270 sessions (where the client supports the RFC 1646 extensions, with the required terminal name separated from the terminal type by an @ sign).

The default is TLNS. The value must be exactly four characters, in PDSNAME format (alpha or national first, and then alpha, national, or numeric for the rest).

The actual names that can be used must be in the format *pppp*####, where *pppp* is the prefix, and #### is a number from 0001 to 9999. (Numbers are allocated by the systems administrator.)

If a requested name is in use, or is not valid, then an error is returned. For TN3270E, it is up to the client to recover. For TN3270, the error is returned as a message on the screen, and the session is closed after a few seconds.

**TOACT1=*action* | (*action,mmm.ss*)**

Specifies the action to be taken for a user when their first terminal time out interval expires. *action* can be: LOG, ALARM, LOCK, DISC, or CANCEL. Optionally, a repeat interval can be specified in the form *mmm.ss* to repeat the specified *action* at that interval. No repeat interval can be specified for an action of DISC, LOCK, or CANCEL. For example:

```
SYSPARMS TOACT1=LOG
SYSPARMS TOACT1=(ALARM,2.30)
```

**Note**

TOACT1 does not take effect until the timeout facility has been enabled by a SYSPARMS TOSTART command. The timeout interval is set using the SYSPARMS TOTIME1 operand.

**TOACT2=*action* | (*action,mmm.ss*)**

Specifies the action to be taken for a user when their second terminal time out interval expires. *action* can be: LOG, ALARM, LOCK, DISC or CANCEL. Optionally a repeat interval can be specified in the form *mmm.ss* to repeat the specified *action* at that interval. No repeat interval can be specified for an action of DISC, LOCK, or CANCEL. For example:

```
SYSPARMS TOACT1=(ALARM,2.30)
SYSPARMS TOACT2=DISC
```

**TOALARM=<u>3</u> | *n***

Specifies the number of times the alarm rings if ALARM is the action specified in the TOACT1 or TOACT2 operand. The default is 3. Valid values are 0 to 9. Specifying 0 suppresses the alarm.

**TODISC=*mmm.ss* | <u>0.00</u>**

Specifies the time out disconnect value. TODISC specifies the interval in minutes and seconds after which a disconnected user is timed out. The default is TODISC=0.00 which means disconnect time out is inactive. TODISC is only active when the user details panel User Timeout option is set to YES.

**TOLOCK=<u>YES</u> | NO**

Specifies whether time out actions (specified in TOACT1 and TOACT2) apply to a locked terminal session.

**TOSTART=<u>NONE</u> | KBUNL | INPUT | OUTPUT | TIO**

Starts or stops terminal inactivity time out processing. Specifying NONE (default) stops time out processing. Specifying another option on this operand starts time out processing according to the following rules:

**KBUNL**

Indicates that terminal inactivity starts from the time the terminal keyboard was last unlocked.

**INPUT**

Indicates that terminal inactivity is measured from the time of the last input received.

**OUTPUT**

Indicates that terminal inactivity is measured from the time of last output to the terminal.

**TIO**

Indicates that terminal inactivity is measured from the time of the last input or output.

In all cases, the point at which terminal inactivity is considered to start represents the start of the TOTIME1 interval for the terminal concerned.

**TOTIME1=<u>0.00</u> | *mmm.ss***

Specifies the interval used as the first level time out period for inactive terminals. Any terminal inactive for this period of time triggers the action defined by the TOACT1 operand. The interval is specified in minutes and seconds in the form *mmm.ss*. Valid values are 0 to 999 minutes and 0 to 59 seconds. The default is 0.00. Specify a zero time interval to suppress all time out processing.

**TOTIME2=<u>0.00</u> | *mmm.ss***

Specifies the interval used as the second level time out period for inactive terminals. Any terminal inactive for this period of time triggers the action defined by the TOACT2 operand. The interval is specified in minutes and seconds in the form *mmm.ss*. Valid values are 0 to 999 minutes and 0 to 59 seconds. The default is 0.00. Specify a zero time interval to suppress second level time out processing.

**TPEND=<u>WAIT</u> | CLOSE**

Specifies the action required by Management Services if its TPEND exit is driven, either because the primary APPL is varied inactive or because VTAM is terminating. Specifying CLOSE terminates Management Services immediately. If WAIT is specified, the system continues operating.

If the SYSPARMS ACBRETRY operand is set to YES, Management Services automatically attempts to reopen its ACB every 30 seconds. If ACBRETRY is set to NO, the system issues a message to the operator and waits for a reply to close down or to reopen its ACB.

**TRACEOPT=[ 06 | *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 default is 06.

For 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’
```

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.
```

The value specified on the TRACEOPT parameter can be any combination of the four values listed above, 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 datastream being written to the activity log. Data recorded can then be examined using the standard on-line log browse facilities.

Tracing is started and stopped by the LUTRACE command.

**UNLOAD=*procname* | (*procname,ddname*)**

Specifies the name of an NCL procedure, or generic procedure name, that is to be unloaded. Any users accessing the procedure can continue to do so, but a warning message is issued to the issuer of the SYSPARMS command. When the procedure is no longer being used, it is deleted from storage.

*ddname* is an optional name, which can be generic, of the DD job control statement that identifies the dataset from which the procedure was preloaded (OS/VS) or the file type (VM). If generic notation is used, CMDREPL procedures which happen to match the generic mask are not unloaded. A generic name is indicated by an asterisk, for example, UNLOAD=($CA*)

As soon as the UNLOAD has been entered, a new copy of the same procedure can be preloaded using the PRELOAD operand. New users of the procedure execute the new copy, even if other users are still using the old copy.

UNLOAD deletes procedures preloaded using the PRELOAD operand, and shared procedures from the NCLPRSHR stack. It can also be used to reverse the earlier replacement of a command by a procedure through the CMDREPL command.

**UNSPVPSZ=YES | NO**

Specifies whether Management Services is to enforce the *unspecified viewport size* BIND option when establishing sessions with display terminals which support the *Read-Partition-Query* command. This operand can only be used if all display terminals that support QUERY can support unspecified viewport size.

The unspecified viewport function sends a BIND to a display terminal without specifying any default or alternate screen sizes. The terminal automatically operates according to its model and screen size characteristics, and Management Services interrogates it for the actual screen sizes it is using. Use of this operand can remove the common errors of network definition associated with specifying logmode table entries.

If this option is used to send a BIND to a terminal that does not support the function, the terminal rejects the BIND. Note that even without this option, logmode table entries for particular terminals can indicate the use of unspecified viewport size and gain similar advantages. This operand is useful in that the logmode entries for terminals need not be changed. See Appendix A, *Connecting Terminals to Management Services*, for further information and examples.

**USERPW=YES | NO | VERIFY**

Specifies whether the NCL system variable &USERPW is available for use in MAI logon data. &USERPW represents the user's Management Services 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.

**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**

Allows users to display only those resources within their defined command partitions. This is the default.

**MSG**

Allows users to 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**

Allows users to display any resource, regardless of the resources specified within their partitions.

**VSAMCAT={ IJSYSCT | *catnum* } (VM systems only)**

Specifies the DLBL name for the VSAM catalog. *catnum* is a 1- to 7-character name. If no CAT operand is specified on the ALLOCATE command, or this SYSPARMS operand is omitted, the default, IJSYSCT (the master catalog) is used.

**VSAMMODE=V | *mode* (for VM systems only)**

Specifies the default mode character used for dataset allocation requests in VM systems. The default is V. *mode* must be a valid VM file mode. If no MODE operand is specified on allocate requests, the VSAMMODE must be used.

**VTAMAUTH=YES | NO**

Specifies whether Management Services is to run using VTAM Authorized Path facilities. Default is YES. Specify NO if authorized path is not to be used.

If your system is configured with the MAI feature, see the description of the SYSPARMS MAIVAUTH operand in your *SOLVE:Access Planning and Installation Guide*.

**VTAMID=NET | *procname***

Specifies the system procedure name used for starting VTAM. *procname* is a 1- to 8-character name. The default with ACF/VTAM and VTAM-G is NET. The default with ECS/VTAM is VTM. If any other procedure name is used, then Management Services must be informed of it so that the correct name can be used when generating VTAM commands.

**VTAMCDRM=NET | *sscpname***

Specifies the name of the VTAMCDRM. This operand is used only for the NTS facility of NetMaster for SNA, where the VTAM SSCPNAME startup option is not set. If using NTS, set VTAMCDRM to the CDRM name for the local VTAM.

**VTAMNTFY=YES | NO (not applicable in VOS3 systems)**

Specifies whether Management Services is to wait for confirmation of acceptance of a CLSDST/PASS function when passing an EASINET terminal to a target application. Specify NO only in VTAM systems which do not support third party notify.

**XABELOW=YES | NO**

Specifies whether Management Services is to allocate buffer storage below the 16 Mb line in XA systems if all storage in the extended private area has been used. The default is NO, meaning that if all XA storage in the Management Services 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 Management Services to run with a severely limited extended private area.

**XNFRETRY=YES | NO (for VOS3 XNF systems only)**

Specifies whether Management Services is to retry the XNF ENTRY process (to catalog the UCE) if the first attempt fails during initialization, or if the connection fails at a later time. If YES is specified, the system retries the ENTRY process every 30 seconds until successful. If NO is specified, a WTOR message is written to the operator console.

**XNFSCRSZ=(*rows, cols*) (for VOS3 XNF systems only)**

Specifies the default screen size for connected terminals. These values, in rows and columns, are used as the alternate screen size in the terminal bind. The default is 24 rows, 80 columns.

**XNFUCENM=*ucename* (for VOS3 XNF systems only)**

Allows you to dynamically change the name of the XNF UCE used by Management Services for XNF connectivity. This operand can be issued only if the UCE is not currently cataloged to XNF. If the UCE is cataloged you receive an error message.

## SYSPARMS Operands for NEWS

NEWS introduces the following operands of the SYSPARMS command for controlling NEWS operation and setting default operational values:

```
SYSPARMS    [ CNMPROC={ procname | FLUSH } ]
            [ CNMACBNM={ acbname } ]
```

**CNMPROC**

Specifies, in 1 to 8 characters, the name of the CNM PROC

**CNMACBNM**

Specifies, in 1 to 8 characters, the name of the CNM ACB

## SYSPARMS Operands for NTS

NTS introduces the following operands of the SYSPARMS command for controlling NTS operation and setting default operational values:

```
SYSPARMS    [ NTSACBNM={ NMNTS | acbname } ]
            [ NTSACCT={ SELECTIVE | ALL | NO } ]
            [ NTSCINTV={ 30 | nn } ]
            [ NTSCLOSE={ YES | NO } ]
            [ NTSCNMQ={ YES | NO } ]
            [ NTSEVENT={ YES | NO } ]
            [ NTSINTSV={ YES | NO } ]
            [ NTSMAIEX={ YES | NO } ]
            [ NTSMAISV={ YES | NO } ]
            [ NTSMAXTP={ 100 | nnn } ]
            [ NTSMAXTR={ 100 | nnn } ]
            [ NTSRSTAT={ YES | NO } ]
            [ NTSRSINT={ 30 | nnn } ]
            [ NTSRSLIM={ 16 | nnn } ]
            [ NTSSAWBF=(n,sK) ]
            [ NTSSKEEP={ 10 | nn } ]
            [ NTSTRBFX={ YES | NO } ]
            [ NTSTRCBF=(n,sK) ]
```

**NTSACBNM={ NMNTS | *acbname* }**

Provides the name of the ACB which NTS opens and uses for both session awareness (or SAW) and session trace (or TRACE) sessions with VTAM. If this operand is not used, the default ACN name of NMNTS is used.

**NTSACCT={ SELECTIVE | ALL | NO }**

Specifies whether NTS accounting data is to be collected for selected sessions only, for all sessions or for no sessions. The default 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.

**NTSCINTV={ 30 | *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 value of 30 seconds should be adequate for most networks.

**NTSCLOSE={ YES | NO }**

Specifies whether any sessions, which remain active when NTS is closed due to system shutdown, are to be considered ended for the purpose of output logging. The default value of 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. Since Management Services 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, Management Services uses the CLOSE operand of the NTS SAW STOP command to similarly close and log all active sessions before stopping.

**NTSCNMQ={ YES | NO }**

Specifies whether NTS is to queue NTS CNM requests. The default, 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 to be purged.

**NTSEVENT={ YES| NO }**

Controls NTS event generation. If you want events to be generated then you must specify SYSPARMS NTSEVENT=YES. Issue SYSPARMS=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.

**NTSINTSV={ YES | <u>NO</u> }**

Specifies whether NTS intensive message logging is active. By enabling this function certain activity can be tracked such as the purging of uncorrelated trace data, dumping of unrecognized entries in the trace buffer etc. This option is not usually required under normal operation.

**NTSMAIEX={ YES | <u>NO</u> }**

Specifies whether, or not, MAI sessions are presented to the NTS user exit. NTSMAIEX=NO is the default and 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. For details of the record format, see the *Implementing the NEWS User Exit* appendix in the *NetMaster for SNA Installation and Administration Guide*.

**NTSMAISV={ YES | <u>NO</u> }**

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.

**NTSMAXTP={ 100 | *nnn* }**

Provides the maximum number *nnn* of trace PIUs which can be specified by any class definition, or operator command, as either the initial or final trace queue depth for any session. The default is 100.

**NTSMAXTR ={ 100 | *nnn* }**

Provides the maximum number *nnn* of resources which 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 in order to collect accounting data. The default is 100.

**NTSRSTAT={ YES | <u>NO</u> }**

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.

**NTSRSINT={ 30 | *nnn* }**

This operand specifies the interval length for resource statistics collection—*nnn* is minutes. . The value of this operand can be in the range 1 through 480 (8 hours). The default value for this operand is 30.

**NTSRSLIM={ 16 | *nnn* }**

Where *nnn* is in the range 0 through 255. This parameter 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 number of intervals is defaulted to 16.

**NTSSAWBF=(*n*,*s*K)**

Is used to specify 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 session awareness data. If not specified a default of 2 buffers of size 4K will be used.

**NTSSKEEP={ 10 | *nn* }**

Specifies the system default session keep count for sessions written to the NTS database. The number *nn* can be from 1 through 255 and 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.

**NTSTRBFX={ <u>YES</u> | NO }**

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 page(s) 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.

**NTSTRCBF=(*n*,*s*K)**

Is used to specify 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.

## SYSPARMS Operands for SYSCMD

This section describes the SYSCMD-specific operands for the SYSPARMS command.

**Function:**

You can initialize or modify the following system parameter values that determine your SYSCMD environment:

```
SYSPARMS   [ AOMCMIGI={ NO | YES  } ][1]
           [ AOMCMIGX={ NO | exitname } ]
           [ AOMCOPTS=xx ]
           [ AOMCPREF={ *DMN | pppp } ][1]
           [ AOMCTO1={ 10 | nnn } ]
           [ AOMCTO2={ 20 | nnn } ]
           [ AOMCTO3={ 5 | nnn } ]
           [ AOMCTYPE={ JES | OP1 | OP2 | OPX | PSEUDO |
                        EXTMCS } ][1]
           [ AOMCUTOK={ NO | YES } ][1]
           [ AOMDBUG2=xx ]
           [ AOMGQSZ={ 80 | 200 | number } ][1 2]
           [ AOMJESCH={ cc...  | NO ]
           [ AOMMLTO={ 5 | number } ]
           [ AOMPQSZ={ 250 | 500 | number } ][1 2]
           [ AOMPRFJI={ YES | NO } ]
           [ AOMPRFJN={ YES | NO } ]
           [ AOMPRFMN={ YES | NO } ]
           [ AOMPRFSN={ YES | NO } ]
           [ AOMPRFTM={ YES | HMS | HM | NO } ]
           [ AOMSSID={ * | NETM | sub-system-id } ][1]
           [ AOMSUBCH={ cc...  | NO ]
           [ AOMTRACE={ YES | NO } ]
           [ AOMTRCRC={ HEX | LIST } ]
           [ AOMTRLIM={ 100 | number } ]
           [ CONIDS=(n, n, a-b, ...)[1]
           [ CONSOLES={ m | (m,p) | 0 | FREE } ][1]
```

**Note**

Operands flagged with [1] cannot be changed while the SYSCMD facility is running.

Operands flagged with [2] have two defaults shown. The first default (smaller value) applies if the operating system *does not* support 31-bit addressing. The second default (larger value) applies if the operating system *does* support 31-bit addressing.

**Use:** The SYSPARMS command and associated operands are used to tailor and control the SYSCMD facility. They are not meant for general use, and their use should be restricted by using command level authority. The following operands apply to all versions of the SYSCMD facility:

**Operands:**

**AOMCMIGI={ NO | <u>YES</u> }**

Sets the default migration ID option. That is, if a SYSCMD command is issued with the MIGID option omitted, and a console is not currently assigned, then the value of this operand setting is used.

This operand cannot be changed when AOM is active or consoles are acquired.

Migration IDs are 1-byte values that allow programs that do not understand console names to communicate with consoles. As most software is in this state, AOMCMIGI=YES is the default.

Only 150 migration IDs are available within one MVS system, so if you are using large numbers of consoles, problems with shortages can arise.

**AOMCMIGX={ NO | *exitname* }**

Allows you to specify a user migration ID determination exit. This exit can be called by your domain to determine if a given command needs a migration ID.

This operand can be changed at any time.

For further details of the migration ID determination exits, see Chapter 2, *Customizing Management Services*.

**AOMCOPTS=*xx***

Sets some fine-tuning for AOM console handling. You can add the values as required. Allocated bits for this operand are:

80 EXTMCS consoles uses name suffixes from 001 to 255 rather than 001 to *m* (from the SYSPARMS CONSOLES definition).
**Note:** The setting of this bit is recognized only when SYSPARMS CONSOLES=*m* is issued. Changing it when a console pool has been acquired has no effect.

01 All MIGID=EXIT requests and AOMMIGID function requests cause a trace message to be written to the log, with details on whether an exit was needed.

This operand can be changed at any time (see note above for value 80).

**AOMCPREF={ *DMN | *pppp* }**

Sets the EXTMCS console name prefix. EXTMCS console names are always in the format Z*ppppnnn* where *pppp* is the prefix (padded to four characters with Z characters), and *nnn* is a numeric suffix. *DMN indicates that the domain ID will be used.

This operand cannot be changed when AOM is active or when consoles are acquired.

**AOMCTO1={ 10 | *nnn* }**

Sets the timeout for a console. It sets the number of seconds during which a console is logically locked following the issue of a command.

This operand applies only if the number of multiline WTO END lines received is greater than or equal to the number of commands issued; that is, only if all commands issued have received an MLWTO response.

The default value is 10. This operand can be changed at any time.

**AOMCTO2={ 20 | *nnn* }**

Sets the timeout for a console. It sets the number of seconds during which a console is logically locked following the issue of a command or receipt of a single-line WTO.

This operand applies only if some commands have had single-line WTO responses, and therefore it is not possible to detect the logical end of output; that is, only if one or both of the following is true:

- The number of multiline WTO END lines received is less than the number of commands issued.
- At least one single-line WTO message has been received.

The default value is 20. This operand can be changed at any time.

**AOMCTO3={ 5 | *nnn* }**

Sets the number of seconds during which the master console is associated with an environment following a SYSCMD CON=MASTER command. The timeout is reset each time a command is issued.

**Note**

Even SYSCMD OPT=REL (which stops message routing to the environment) does not reset this timeout. The master console is not available to other SYSCMD OPT=MASTER issuers until the time expires.

The default value is 5. This operand can be changed at any time.

**AOMCTYPE={ JES | OP1 | OP2 | OPX | PSEUDO | EXTMCS }**

Sets the type of console to be used.

The default value depends on the operating system:

| | |
|---|---|
| JES | MVS through to MVS/ESA |
| EXTMCS | OS/390 and above |
| OP1 | MSP and above |
| PSEUDO | VOS3 |

The allowable values depend on the operating system:

| | |
|---|---|
| JES | MVS through to OS/390 and beyond and MSP |
| EXTMCS | MVS/ESA 4.2 through to OS/390 and beyond |
| OP1, OP2, OPX | MSP and above |
| PSEUDO | VOS3 |

The meanings of the values are:

| | |
|---|---|
| JES | On MVS or MSP, use JES consoles. |
| EXTMCS | On MVS/ESA 4.2 or later, use Extended MCS consoles. |
| OP1 | On MSP, use OP1 consoles. |
| OP2 | On MSP, use OP2 consoles. |
| OPX | On MSP, use both OP1 and OP2 consoles. |
| PSEUDO | On VOS3, use PSEUDO consoles. |

This operand cannot be changed when AOM is active or when consoles are acquired.

**AOMCUTOK={ NO | YES }**

Sets the option for consoles to use SAF UTOKENs. This is possible only with MVS/ESA or OS/390, and 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.

**AOMGQSZ={ 80 | 200 | *number* }**

Specifies the global queue size in buffers. Each buffer holds one message and takes 416 bytes. Value must be in the range 5 to 500 inclusive.

The global queue is a collection point for all system messages destined for the system. Because messages are regularly cleared from the global queue and placed in buffers in the private queue (which has its size set by SYSPARMS AOMPQSZ), the global queue size should be a smaller value than the private queue; the private queue buffers taking longer to clear.

The actual size of the global queue can be greater than the value specified on this operand. This is because the total queue size is rounded up to a page boundary.

The higher default, 200, applies in a 31-bit addressing environment.

**AOMJESCH={ *cc* | NO }**

Set the JES command recognition character(s) in use on this system. Normally, for JES2, the character is the dollar ($), and for JES3, the character(s) are the asterisk (*), and 8. However, these can be changed. To allow the AOM migration ID determination exit to correctly detect JES commands, if non-standard characters are being used, use this SYSPARM to indicate the character(s) in use.

Setting AOMJESCH=NO clears the characters, and the default characters are recognised. Otherwise, you can specify a value of one or two characters from the following set:

$ * 8 ? / ¢ ! + @ # % & < > , . ; : - _ = |

This operand can be changed at any time.

**AOMMLTO={ 5 | *number* }**

Specifies the maximum time that the SYSCMD facility will allow for any one multiline WTO to be completely collected. If this timeout 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.

The allowable range is 2 to 60 seconds.

**AOMPQSZ={ 250 | 500 | *number* }**

Specifies the private queue size in buffers. Each buffer holds one message and takes 416 bytes. Value must be in the range of 5 to 10000 inclusive.

The private queue is *owned* by NetMaster for SNA and is the distribution point for all messages destined for the system, received from the global queue. Because messages in the private queue take longer to process, the private queue should be set to a larger value than the global queue.

The actual size of the private queue can be greater than the value specified on this operand. This is because the total queue is rounded up to a page boundary.

The private queue is allocated in NetMaster for SNA private storage (above the 16M line in MVS/ESA and OS/390).

The higher default, 500, applies in a 31-bit addressing environment.

**AOMPRFJI={ YES | NO }**

Specifies if the JES job ID is to prefix the message text when displayed at an OCS screen, for messages sourced by an MVS system. The value of this parameter is used to set the message prefixing actions when a new OCS or NCL &INTCMD environment is created.

**AOMPRFJN={YES | NO }**

Specifies if the MVS job name is to prefix the message text when displayed at an OCS screen, for messages sourced by an MVS system. The value of this parameter is used to set the message prefixing actions when a new OCS or NCL &INTCMD environment is created.

**AOMPRFMN={YES | NO }**

Specifies if the minor lines of MVS-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.

**AOMPRFSN={ YES | NO }**

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.

**AOMPRFTM={YES | HMS | HM | NO }**

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*.

**AOMSSID={ * | NETM | *sub-system-id* }**

Specifies the subsystem ID to be used by SYSCMD facility. The asterisk (*) uses the first four characters of the NetMaster for SNA job name or started task (STC) name.

**AOMSUBCH={ *ccc*… | NO }**

Sets an optional list of subsystem command prefix characters. The AOM Migration ID determination exit recognizes these characters as indicating commands for which no migration ID is required.

Setting AOMSUBCH=NO clears the characters. Otherwise, you can specify a value of 1 to 16 characters from the following set:

$ * ? / ¢ ! + @ # % & < > , . ; : - _ = |

This operand can be changed at any time.

**AOMTRACE={YES | <u>NO</u> }**

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.

**AOMTRCRC={ <u>HEX</u> | 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.

**AOMTRLIM={ <u>100</u> | *number* }**

Specifies the tracing limit. When reached, AOMTRACE is set to NO. Value must be in the range 1 to 100 inclusive.

**CONIDS=(*n,n,a-b,*…)**

In an MSP environment only, specifies the JES, OP1, or OP2 console IDs that are to be used by AOM.

Entries in the list can be either a single console ID (*n*), or a range (*a-b*). During acquisition, only the specified console IDs are considered. They must also match the specified console type, as set by SYSPARMS AOMCTYPE (JES, OP1, or OP2).

This operand cannot be changed when AOM is active or when consoles are acquired.

**CONSOLES={ *m* | (*m*,*p*) | 0 | FREE }**

Acquires and releases consoles.

CONSOLES=*m* or CONSOLES=(*m*,*p*) is used to initialize the console pool.

*m* is the maximum number of consoles that can be used by this region at any instant. *m* can be from 1 (for a SYSCMD system) or 2 (for a full AOM system) to 254.

*p* is the pool number. It is the maximum number of consoles that will be left free as consoles that are being used are released. If *p* is specified, it must be in the range 0 to *m*. If *p* is not specified, then it defaults as follows:

- For *m* less than or equal to 5, *p* equals *m*.
- For *m* greater than 5 and less than or equal to 15, *p* is 5.
- For *m* greater than 15, *p* is *m*/3.

The CONSOLES=0 or CONSOLES=FREE option is used to release the console pool. Any allocated consoles are released back to MVS.

This operand cannot be changed when AOM is active.

# E

# UAMS Panel Descriptions

This appendix describes the fields on each of the UAMS panels that are used to define a user ID. Use these descriptions as a guide for deciding the authority each user is assigned.

Fields described as mandatory must be entered and not left blank. Optional fields can be left blank.

## The User Details Panel

The fields on the User Details panel are described below:

**UserID (mandatory)**

Specifies the user ID you are defining. This is a 1 to 8 alphanumeric or national character string (for example a Kanji character).

**Initial Password (optional—Add processing only)**

Specifies the initial password for the user ID you are defining. The length of this password is determined by the SYSPARMS PWMIN and PWMAX operands (see Appendix D, *SYSPARMS Operands*, for more information).

**Note**

This field should not be specified if a group ID is being defined.

The new user must be notified of their initial password. This password only allows the very first logon of the new user ID. The system then forces the new user to provide their own password before they can perform any functions.

If your system is configured with a partial security exit that provides password processing, this field can be left blank.

**User Name (mandatory)**

Specifies the name of the owner of the new user ID, or any other name required to identify who uses this user ID.

**User Location (optional)**

Specifies the location, site, business address, and so on, details for the user. This information is used in later displays to allow easy correlation with the associated user ID.

**Phone Number (optional)**

Specifies the user's telephone number(s).

**Language Code (optional)**

Specifies the 2-character language code chosen by your installation. If left blank this field is set to the value of the SYSPARMS LANG operand (see Appendix D, *SYSPARMS Operands*, for more information).

This field value is made available to NCL procedures in the &ZUSERLC system variable.

**Time Zone Name (optional)**

Specifies the user's time zone name. If not specified, it defaults to SYSTEM. This field is validated by the $UA table TZNAME. All valid time zone names must be added to this table. See the *MODS Programming and Administration Guide* for details of how to add time zone names to the TZNAME table.

**Suspend Date (optional)**

Specifies the date (format YY.DDD) on and after which this user ID is refused access to the system. If the current date is specified, the user ID is immediately disabled.

**User Timeout**

Specifies whether this user ID is subject to the timeout management facility. Valid values are Y or N. If Y is entered, timeout applies whenever the user ID is logged on but has been inactive for a specified period. See Chapter 2, *Customizing Management Services*, for information on the timeout facility.

**Group ID**

Specifies the group ID that applies to this user ID. Specify this field if the user ID is to be a member of a Group of users, all of whom have the same attributes. The group ID definition must already exist.

**Note**

If a Group ID is specified, one other panel can be displayed—press F8 to change the Event Notification Services user attributes and then F3 to add (file) the new user ID.

**Group Name**

Specifies the user name of the group ID, if a group ID is specified.

**Last Logon**

Specifies the time and date of the last successful logon for this user. This is blank for Add processing.

**Last Updated**

Specifies the time and date of the last update of this user definition. This is blank for Add processing.

**Definition**

Specifies the type of definition being browsed or maintained. Valid values are User (for an individual user definition) or Group (for a group definition).

**Initial OCS Command**

Specifies the command text that is to be executed automatically on entering OCS mode. This field can be modified by each user by issuing a PROFILE command.

If this is a remote user ID definition, the value of this field is executed when the user signs on to the remote system using the ROF SIGNON command, except when the command is issued from an NCL procedure using the &INTCMD facility.

**Note**

This field is only displayed here for a user who is a member of a group (Group ID specified), provided that group has OCS authority. To display this field for input, press the ENTER key after entering the group ID name.

## The User Authorities Panel

The User Authorities panel is displayed after the User Details panel. It allows you to specify details of the access the user has to system functions. The fields on this panel are as follows:

**Authority Level (mandatory)**

Specifies the command authority level for the user ID. The authority level defines which Management Services (MS) commands they can use. Valid values are 0 (lowest authority) to 255 (highest authority).

When allocating command authorities, consider the way the command will be used and who will use it. Default authorities are allocated according to the following guidelines:

**0** Monitoring and display commands only

**1** Network management commands

**2** Commands with special operands that have global impact

**3** Major impact commands, including system shutdown

**4** Commands, (and certain high-impact operands of lower authority commands), reserved for the systems programmer, such as commands used for problem determination or performance tuning

A range of 0 to 10 is adequate for most installations. Operational staff are normally allocated authority level 3, while levels 4 and 5 are allocated to systems programming staff responsible for system installation and maintenance.

See Appendix H, *Command Authority Levels*, for a description of the commands available and their default authority levels. For information on changing the authority levels of individual commands, see Chapter 2, *Customizing Management Services*.

**Multiple Signon Authority (mandatory)**

Specifies whether the user ID is allowed to have multiple concurrent terminal logons to the same system. Valid values are Y and N. The default is N. If N is specified, the user can have a native 3270 session, a ROF session, and a TSO session to the same system at the same time, but not more than one of each type.

**Split/Swap Authority (mandatory)**

Specifies whether the user ID is allowed to use the screen window split and swap facilities. Valid values are Y and N. If N is specified, the F2 (Split) and F9 (Swap) keys are disabled and the user ID is not allowed to use OCS SPLIT or SWAP commands.

**Panel Command Entry Authority**

Specifies whether the user ID is allowed to enter the EXEC and CMD commands in the COMMAND ===> and Select Option ===> fields on panels. Valid values are Y or N. If the user has OCS authority, this field is ignored, and the user is assumed to have this authority.

**Password Expire (mandatory)**

Specifies whether the user ID is subject to automatic password expiry at the frequency chosen by your installation. Valid values are Y or N. The frequency of expiry is defined by the SYSPARMS PWEXPIRE operand. If N is specified, the user is not forced to change the user ID password. If you have a security exit which controls password expiry, this field is overridden.

**TSO Autologon**

Specifies whether the user ID can be used to log on to a system without logging off TSO. Valid values are Y or N. The default is N. The user ID must have the same name as the TSO user ID.

If N is specified, the user must enter their Management Services user ID and password to proceed with the session.

If Y is specified, the system allows them to log on directly (without prompting for their user ID and password), on the basis that their logon to TSO is already sufficient proof of identity.

**Note**

If you require users to log on to systems executing under different operating systems, such as VM, the TSO AUTOLOGON attribute must still be set for the user's VM system user ID, if automatic logon from their MVS TSO system is to be allowed by the VM system.

**APPC Access Key (optional)**
**APPC Access Lock (optional)**

Specifies the authorization for one user to act on behalf of another user, without supplying the target user's password. The default for both the APPC Access Key and APPC Access Lock fields is NONE.

Both the APPC Access Key and APPC Access Lock represent 16 bit fields, equivalent to, from left to right, the access codes 1 to 16.

When one user needs to act as another user, the requesting user's access key is compared with the target user's access lock. Access is denied through an access code if the indicating bit for an access code is off (set to zero) in the requestor's access key. Access is granted if the indicating bit in the target user's access lock is on (set to one), *only* if the corresponding access code bit in the target user's access lock is on.

If access is granted through any access code, the requesting user is authorized to act as the target user without supplying a password.

Therefore an access key of NONE (all zeroes) prevents any attempt by that user to act as another, while a key of ALL (all ones) would allow that user to act as any other user that had at least one access code indicator bit set in the access lock. Likewise, an access lock of NONE prevents any access to that user ID without supplying a valid password.

Valid entries for both the APPC Access Key and APPC Access Lock fields are as follows:

- Enter ALL to set all indicator bits to one
- Enter NONE to set all indicator bits to zero
- Enter a list of access code numbers which indicate the bits which are required to be set on, separated by commas, or a range of access codes. For example, entering 2,5,10-12,16 will set the field to 0100100001110001.

**NCL Library DDNAME (optional)**

Specifies an alternative NCL procedure library for the user ID. When the user executes an NCL procedure the system searches this library for the procedure.

You can assign a procedure library by entering the DD name (OS/VS) or the file type (VM systems) for the alternative library in this field.

If left blank, the library is the one identified by the COMMANDS DD card in the execution JCL. In OS/VS systems, the library is that defined in the system COMMANDS dataset. For VM systems, the library is that defined by the NCLFT or NCLFM operands.

**User Services Member (optional)**

Specifies whether the user ID has access to Management Services User Services facilities. The name of an NCL procedure for this user ID must be specified. If no procedure name is provided, the user cannot access User Services. The default is $USERSER, which is the name of a sample User Services NCL procedure distributed with the system.

**Panel Library Path Name (Optional)**

Specifies the name of the path definition that is to be used when panels are retrieved on behalf of the user ID. The default path name is PANELS. See Chapter 3, *Implementing Security*, for more information on path definition.

**Terminals to which User will be restricted (Optional)**
**Terminal 1, Terminal 2, Terminal 3**

Specifies the VTAM logical unit (node) names for one, two, or three terminals to which the user ID is restricted. The system refuses any attempt to log on at any other terminal by this user ID. This facility is useful for restricting the activity of certain high authority user IDs to specific terminals and specific locations. If all fields are left blank, no restriction is placed on the origin of the user ID logon.

## The Access Authorities Panel

This panel defines the user's access authority for the major products supported by Management Services. If your system is not configured for some of the optional products listed, the associated fields are not be available for maintenance.

**Network Management**

Specifies whether the user ID has access to network management facilities. Valid values are Y or N. The default is N. Enter Y to allow access to the Network Management option from the Primary Menu. If Y is entered, an additional Network Management panel is displayed during the user ID addition process.

**Systems Administration**

Specifies whether the user ID has access to SOLVE:Central applications. Valid values are Y or N. The default is N. Enter Y to allow access to the Systems Administration option from the Primary Menu. If none of the applications are licensed, this field cannot be set to Y.

See the *SOLVE:Central Implementation and Administration Guide* for further information about SOLVE:Central applications.

**Information Management**

Specifies whether the user ID has access to INFO/MASTER

Valid values are Y or N. Enter Y to allow access to the Information Management option from the Primary Menu.

**INFO/MASTER Maintenance**

Specifies whether the user ID is to be allowed maintenance privileges for INFO/MASTER category definitions. Valid values are Y or N. The default is N.

**Operations Management**

Specifies whether the user ID has access to Automation Services.

Valid values are Y or N. The default is N. If your system is not configured with products that require Automation Services, this field is not available for maintenance.

If Y is entered, additional Automation Services, AOM, and SYSCMD details panels are displayed later in the addition process. See your *Automation Services Administrator Guide* for details of these additional panels.

For details on access authorities for Automation Services and AOM, see the *Automation Services Administrator Guide*.

**Access Services**

Specifies whether the user ID has access to SOLVE:Access Multiple Application Interface/Full Screen (MAI-FS). Valid values are Y or N. The default is N. If the system is not configured with this feature, this field is not available for maintenance.

Enter Y to allow access to the Access Services option from the Primary Menu. If Y is entered, an additional MAI Details panel is displayed later in the addition process. See the *SOLVE:Access Planning and Installation Guide* for details about this panel.

**Data Transfer Services**

Specifies whether the user ID has access to File Transfer Services (FTS). Valid values are Y or N. The default is N. If your system is not configured with FTS, this field is not available for maintenance.

Enter Y to allow access to the Data Transfer Services option from the Primary Menu. If Y is entered, an FTS Details panel is displayed at a later stage in the addition process. See your *FTS Planning and Installation* guide for details about this additional panel.

**Electronic Mail**

Specifies whether the user ID has access to SOLVE:Netmail. Valid values are Y or N. The default is N. If SOLVE:Netmail is not licensed, this field cannot be set to Y. Enter Y to allow access to the Electronic Mail option from the Primary Menu.

**Operator Console Services**

Specifies whether the user ID has access to Operator Console Services (OCS). Valid values are Y or N. The default is N. Enter Y to allow access to the Operator Console Services option from the Primary Menu, and to allow this user to act as an OCS operator. If Y is entered, an OCS Profile panel is displayed at a later stage in the addition process.

**Management Services**

Specifies whether the user ID has access to Management Services. Valid values are Y or N. The default is N. Enter Y to allow access to the Management Services option from the Administration Menu (/ADMIN). If Y is specified, restrictions for the following Management Services functions can also be specified.

**UAMS Maintenance**

Specifies whether the user ID has authority for the UAMS function. It is your responsibility to determine which users are allowed access to this function.

**Broadcast Services**

Specifies whether the user ID has access to broadcast services. You may need to restrict authorization for this function.

**Object Services Support**

Specifies whether the user ID has access to SOLVE:Central Object Services Support functions.

**Object Services Security**

Specifies whether the user ID has access to the SOLVE:Central Object Services Security functions.

**System Support Services**

Specifies whether the user ID has access to the System Support Services function. This function is usually reserved for Systems Programmers or Operations Support personnel.

**Managed Object Dev. Services**

Specifies whether the user ID has access to Managed Object Development Services. Valid values are Y or N. The default is N. Enter Y to allow access to the Managed Object Development Services option from the Management Services Primary Menu (/MS).

If Y is specified, the user ID has browse access to all options on the Common Application Services (CAS) menu, the Application Register, and Object Class Specification. Update access is provided by specifying Y in the MODS Details panel entries (see *The MODS Details Panel*, on page E-10).

When all access authorities have been specified, press F8 to scroll forward to the MODS Details panel.

## The MODS Details Panel

The MODS Details panel is presented only if the user's Managed Object Services field is set to Y. The fields on this panel are as follows:

**Application Register**
Specifies whether the user ID has update access to the MODS Application Register. Valid values are Y or N. The default is N.

**Menu Maintenance**
Specifies whether the user ID has update access to CAS Menu Maintenance. Valid values are Y or N. The default is N.

**List Maintenance**
Specifies whether the user ID has update access to CAS List Maintenance. Valid values are Y or N. The default is N.

**Panel Maintenance**
Specifies whether the user ID has access to MODS Panel Maintenance. Valid values are Y or N. The default is N.

**Panel Domain Maintenance**
Specifies whether the user ID has update access to CAS Panel Domain Maintenance. Valid values are Y or N. The default is N.

**Help Maintenance**
Specifies whether the user ID has update access to CAS Help Maintenance. Valid values are Y or N. The default is N.

**Message Maintenance**
Specifies whether the user ID has update access to CAS Message Maintenance. Valid values are Y or N. The default is N.

**Table Maintenance**
Specifies whether the user ID has update access to CAS Table Maintenance. Valid values are Y or N. The default is N.

**Criteria Maintenance**
Specifies whether the user ID has update access to CAS Criteria Maintenance. Valid values are Y or N. The default is N.

**Command Maintenance**
Specifies whether the user ID has update access to CAS Command Maintenance. Valid values are Y or N. The default is N.

**Object Class Specification**
Specifies whether the user ID has update access to OS Object Class Specification. Valid values are Y or N. The default is N.

**Mapping Services**

Specifies whether the user ID has access to Mapping Services. Valid values are Y or N. The default is N.

**Administration**

Specifies whether the user ID has access to the administration functions of MODS. Valid values are Y or N. The default is N.

## The Installation Attributes Panel

This panel allows you to specify ten optional installation attributes. These fields can contain any information in free form text. These fields are accessible from NCL (see the &SECCALL verb definition in the *Network Control Language Reference*). They allow you to link installation specific information with a user definition.

When complete, press F8 to scroll forward to the next panel. This is the OCS details panel if you are defined with OCS access, otherwise it is the Notification Details panel.

## The OCS Details Panel

The OCS Profile panel is displayed if Y was entered against the OCS access option on the Access Authorities panel. This panel allows entry of the privileges and authorities associated with the user's access to the OCS facility. The following fields appear on this panel:

**Monitor Status (mandatory)**

Specifies whether the user ID has Monitor status. Values are Y or N. The default is N.

Monitor status user IDs receive notification of events happening within the system that is not directly associated with their user ID. They are notified of certain unsolicited events throughout the network, such as other users logging on and off, and security violations. This function is normally allocated to network operations staff and any other personnel directly connected with monitoring activity within the network.

**Receive PPO Messages (mandatory)**

Specifies whether the user ID is entitled to receive unsolicited VTAM Primary Program Operator (PPO) messages. Values are Y or N. The default is N.

The user can suppress the receiving of these commands by using the PROFILE command. The range of the PPO messages delivered to a user can be specified with message Network Partitioning.

**Message Code (optional)**

Specifies the user ID's message code. This message code is a 2-digit hexadecimal number representing an 8-bit string. Valid values are 00 to FF. The default is 00.

The message code attribute allows an installation to further classify which users are to receive particular messages. If an &WRITE NCL statement is issued specifying the MSGCODE operand, the message is sent to a user only if it matches the MSGCODE value on the statement (also expressed as a bit string), or if it is a subset of the user's message code bit string. See the &WRITE verb description in the *Network Control Language Reference* for more information.

The AOM component of Management Services also allows MSGCODE values to be assigned to messages. See your AOM manuals for more information.

**Message Procedure (optional)**

Specifies the name of an NCL procedure operating as the MSGPROC for this user ID. If no member name is specified, standard message delivery applies to the user ID.

**Minimum Severity Class (mandatory)**

Specifies a severity level for unsolicited (PPO) VTAM messages that can be displayed on the user's terminal. The codes for the severity levels are the same as those used in the VTAM messages manual. The severity level entered specifies that messages at or above that level are displayed.

This field is only valid if the earlier Receive PPO Messages field has been set to Y. Otherwise disregard all other fields except INITCMD at the bottom of the panel.

**NPF Resource List Member (optional)**

Specifies the name of an NPF dataset member that specifies resource table names that are to be used for resource filtering for this user. This member allows the combination of message and command resource members as well as user defined resources. The NPF Command Member and NPF Message Member fields are cleared and ignored if any entry is made in this field.

**NPF Command Member (optional)**

Specifies the name of an NPF control member specifying command partitioning for the user ID. This field cannot be specified if a Resource List Member is specified.

See Chapter 8, *Partitioning VTAM Networks*, for more information on command partitioning.

**NPF Message Member**

Specifies the name of an NPF control member specifying message partitioning for the user ID. This field is applicable if the Receive PPO Messages field is YES. Otherwise it is ignored. This field cannot be specified if a Resource List Member is specified.

See Chapter 8, *Partitioning VTAM Networks*, for more information on message partitioning.

**Undeliverable PPO Messages (mandatory)**

Specifies whether the user is to receive PPO messages that cannot be delivered to any other user despite the user ID's NPF restrictions. Applicable only if the user is authorized to receive PPO messages, and if an NPF control member for message partitioning has been specified.

**Unsolicited message receipt (mandatory)**

Specifies the default value for the PROFILE UNSOL command when entering OCS. See the *Management Services Command Reference* for a complete description of the PROFILE command.

**MSG message receipt**

Specifies the default value for the PROFILE MSG command when entering OCS. See the *Management Services Command Reference* for a complete description of the PROFILE command.

**Initial OCS Command (optional)**

Specifies the command text that is to be executed automatically on entry to OCS mode. This field can be modified online using the PROFILE command. This command is also executed when a remote user ID signs on to the remote system using the ROF SIGNON command, except when the command is issued from an NCL procedure using the &INTCMD facility.

**Note**

This field is ignored if you are defining a group. Users defined as members of a group can set their own initial OCS commands.

## The Notification Details Panel

The Notification Details specified on this panel are used by various applications to notify a user of an event. The fields on this panel define the mode or method by which the user wishes to be notified. They also define what mode or method they are notified by, depending on the time of day.

The notification details defined for a user ID also affects the sending of broadcasts. See the *Management Services User's Guide* for details.

The fields on the Notification Details panel are as follows:

**Notify by Mode**

Specifies the notification mode or method to be used if the day of week and time of day match. Specify a question mark (?) to obtain a selection list of valid modes. Valid notification modes are shown in Table E-1.

*Table E-1.Notification Modes*

| Identifier | Notification Mode |
|---|---|
| BC | Broadcast message, available on any system |
| BCL | Broadcast message (saved if the user is not logged on), available on any system |
| CMS | VM/CMS message, available on any VM system or through an INMC link to a VM system (the target domain must run VM) |
| EXIT | User supplied notification exit, can use whatever facilities are available to communicate throughout the network |
| MEMO | Memo mail system (see the *SOLVE:Central Implementation and Administration Guide* for more information) |
| MSG | OCS message, available on any system or through ROF to another system |
| NETMAIL | NET/MAIL (V2.2.2 or later), within the same system or through INMC |
| NETMAIL1 | NET/MAIL (prior to V2.2.2), within the same system or through INMC |
| TSO | TSO/TSS Message, available on any MVS system or through an INMC link to an MVS system (the target domain must run MVS) |

*Table E-1.Notification Modes*

| Identifier | Notification Mode |
|---|---|
| TSOL | TSO/TSS Message (saved if the user is not logged on), available on any MVS system or through an INMC link to an MVS system (the target domain must run MVS) |
| NONE | Notification message is discarded |

**Domain**

Specifies the domain at which the user is to receive the notification. The following rules apply:

- The system receiving the notification request must have the Background System user of the system sending the request defined.
- The sending system must have an active link established between itself and the receiving system.
- The existence of the domain is only validated at the time a notification is sent.

**SMTWTFS**

Specifies a string of Ys and Ns that correspond to the seven days of the week. If a Y is present for a particular day then the specified notification rule is able to be used if a notify request is issued on that day. An N indicates that the rule cannot be used on that day of the week.

**Times**

Specifies the start and end times during which a notification rule applies to a notify request (provided the day of week has passed the screening). The two times are inclusive and are of the format HHMM (where HH is in a 24 hour format and MM are minutes). The times can be specified in the standard shorthand ways, for example, .2 will be translated into 0020 and 5.3 will be translated to 0530 (note, the '.' character will be removed from the time).

**User ID**

Specifies the user ID that the notification is to be sent to. If not specified, the user ID of the definition is used. This allows users who have different user IDs on different systems to receive notifications for these user IDs.

**Note**

No verification is performed on the user ID that is specified.

**Exit Name**

Specifies an NCL procedure that contains installation defined notification methods. This procedure must exist in the systems that are to receive the notification. For more information on coding a notification exit see the *Network Control Language Reference*.

**Parameters**

Specifies parameters that are passed to the notification exit (if it is specified).

## The Network Management Details Panel

If Y was entered in the Network Management Access field on the Access Authorities panel, the Network Management panel is displayed.

The Network Management panel contains the following fields:

**NEWS Access (mandatory)**

Specifies whether the user ID has access to the Network Error Warning System (NEWS) of NetMaster for SNA. Valid values are Y or N. The default is N.

**Reset Authority (mandatory)**

Specifies whether the user ID has authority to delete NEWS database records of NetMaster for SNA. Valid values are Y or N. The default is N.

**NTS Access (mandatory)**

Specifies whether the user ID has access to the Network Tracking System (NTS). Valid values are Y or N. The default is N. NTS access requires the system to be configured with NetMaster for SNA.

**NCS Access (mandatory)**

Specifies whether the user ID has access to the Network Control System (NCS). Valid values are Y or N. The default is N.

**NCPView Authority**

Specifies the level of access the user ID has to NCPView. Valid values are 0 (not authorized to access NCPView) or 1 (authorized to browse). The default is 0. This requires NetMaster for SNA.

**TCP/IP Services**

Enter 1 to allow the user view only access to TCP/IP Services, or 2 to allow view and update access.

## Print Services Manager Details Panel

This panel describes the privileges available to a user associated with the Print Services Manager (PSM) facilities of Management Services. This panel is presented for all users, regardless of their access authorities. See your *Management Services User's Guide* for details of Print Service Manager (PSM) operations.

**Primary Menu Access (optional)**

Specifies whether the user ID has access to Print Services Manager (PSM). Valid values are:

**0** The user is not authorized to access PSM (the default).

**1** The user is authorized to access PSM.

**Maintenance Access (optional)**

Specifies the level of access the user ID has to the maintenance functions of PSM. Maintenance functions include printer, form, and setup definitions as well as printer assignments. Valid values are:

**0** The user is not authorized to access the maintenance functions of PSM (the default).

**1** The user has view only access of the maintenance functions of PSM.

**2** The user has add and update access to the maintenance functions of PSM.

**3** The user has add, update, and delete access to the maintenance functions of PSM.

**Administration Access (optional)**

Specifies the level of access the user ID has to the administration functions of PSM. Valid values are:

**0** The user is not authorized to access the administration functions of PSM (the default).

**1** The user has view and update access of the administration functions of PSM.

**2** The user has view and update access to the administration functions of PSM as well as authority to clear the print spool.

**Ability to Change Priority (optional)**

Specifies whether the user ID has authority to change the priority of print requests. Valid values are:

**0** The user is not authorized to change the priority of print requests (the default).

**1** The user is authorized to change the priority of their own print requests.

**2** The user is authorized to change the priority of all print requests.

**Queue Access (all) (optional)**

Specifies the level of access the user ID has to the output queue of other users. Valid values are:

**0** The user is not authorized to view the output queue when other user's output is present (the default).

**1** The user is authorized to display the output queue when other user's output is present.

**2** The user is authorized to browse other users' output.

**3** The user is authorized to display, browse, modify, release or hold other users' output.

**4** As for 3, with authority to delete other users' output .

**Queue Access (own)**

Specifies the level of access the user ID has to their own output queue. Valid values are:

**0** The user is not authorized to display the output queue when their own output is present (the default).

**1** The user is authorized to display the output queue when their own output is present.

**2** The user is authorized to browse their own output.

**3** The user is authorized to display, browse, modify, release or hold their own output.

**4** The user is authorized to display, browse, modify, release or hold their own output.

## Report Writer Details Panel

This panel describes the privileges that the user has associated with the Report Writer facility of Management Services. This panel is presented for all users, regardless of their access authorities.

**Primary Menu Access (optional)**

Specifies whether the user ID has access to Report Writer. Valid values are:

**0** The user is not authorized to access Report Writer (the default).

**1** The user is authorized to access Report Writer.

**Administration Access (optional)**

Specifies whether the user ID is authorized to perform Report Writer table maintenance functions. Valid values are:

**0** The user is not authorized to perform Report Writer table maintenance functions (the default).

**1** The user is authorized to perform Report Writer table maintenance functions.

**Report Maintenance Access (optional)**

Specifies whether the user ID has access to Report Definition Maintenance. Valid values are:

**0** The user is not authorized to access to Report Definition Maintenance (the default).

**1** The user is authorized to access to Report Definition Maintenance.

**Public Report Access (optional)**

Specifies the level of access the user ID has to public reports. Valid values are:

**0** No access is allowed (the default).

**1** Generation, copying, and viewing of public reports is allowed.

**2** As for 1, with update authority

**3** As for 2, with delete authority

**Private Report Access (Own) (optional)**

Specifies the level of access the user ID has to their own private reports. Valid values are:

**0** No access is allowed (the default).

**1** Generation, copying and viewing of their own private reports is allowed

**2** As for 1, with update authority

**3** As for 2, with delete authority

**Private Report Access (All) (optional)**

Specifies the level of access the user ID has to other users' private reports. Valid values are:

**0** No access is allowed (the default).

**1** Generation, copying and viewing of other user's private reports is allowed.

**2** As for 1, with update authority

**3** As for 2, with delete authority

**Schedule Maintenance Access (optional)**

Specifies the level of access the user ID has to Schedule Definitions. Valid values are:

**0** No access is allowed (the default).

**1** The user is authorized to view Schedule Definitions.

**2** The user is authorized to view, add, or update Schedule Definitions.

**3** As for 2, with delete authority

**4** As for 3, with authority to stop and start schedule processing

# F

# User ID Security Exit Support

This appendix provides information on how to code the user ID security exit and provides the parameter lists for all the user ID security exit calls.

## External Security Packages

External security packages, such as RACF or CA-ACF2, can provide minimal or full security checking.

If your organization has an external security package, access to that package is provided through one of the following types of exit:

- Partial security exit—password and logon access maintenance is controlled by the external security package while UAMS stores the user definitions.
- Full security exit—all security functions are maintained and stored by your external security package.

The following sample security exits are provided and can be found in the following libraries:

- The SMP target zone library, *dsnpref*.INSTAL
- The SMP distribution zone library, *dsnpref*.BASE.INSTALL

| Sample Exit | Description |
|---|---|
| CCRACFFX | Full security exit for RACF |
| CCACF2FX | Full security exit for CA-ACF2 |
| NMSAFPX | SAF partial security exit |

**Note**

The SMP target zone is updated when you SMP APPLY maintenance, whereas the SMP distribution zone is updated only when you subsequently SMP ACCEPT the maintenance. To ensure that you include all applied maintenance, it is recommended that you use the member in the SMP target zone, that is, in *dsnpref*.INSTAL.

The NMSAFPX partial security exit is a SAF-based security exit that supports UTOKENS. SAF is the IBM System Authorization Facility and is the agreed standard for the encoding of requests that require security checking. To assemble and link-edit your security exit, use sample JCL member NMSAFPXL, which is in the same dataset(s) as the NMSAFPX sample exit.

**Note**

SAF is documented in IBM's *Security Server RACROUTE Macro Reference* manual. Refer to the documentation for your security package to see whether the package supports SAF-formatted calls.

For OS/390 and z/OS systems, it is recommended that you use the NMSAFPX partial security exit as it supports RACF and CA-ACF2 security packages.

When the exit has been written, link it to create a load module. Place the module in the load library, or in another library concatenated to the load library through the STEPLIB DD statement in the started task JCL. Change the Management Services JCL parameters to include SEC={*|NO|PARTSAF|*name*}. This causes the security exit to be used.

## Writing Your Own User ID Security Exit

The security exit is coded as an assembler language module, or suite of modules. These modules must be able to:

- Accept the various parameter lists passed to it by Management Services
- Return the designated return codes associated with those parameter lists

When the exit has been written, you link it into a standalone load module. The name of this load module is arbitrary (for example, MYSECURX). The load module is placed in an APF authorized library or another library concatenated to the Management Services load library in the JCL STEPLIB DD statement.

Management Services determines the type (partial or full security) of exit being provided. It does this by examining the data at the entry point of the exit load module.

The entry point of the security exit load module must look like this:

```
ENTRY  B    SKIP-ENTRY(,R15)   Skip descriptor
       DC   ALI 'S'            Required, constant 'S'
       DC   ALI (0)            Required, interface version,
                                         only 0 supported
       DC   ALI (type)         Required, exit type,
                                         0: partial
                                         4: full
       DC   ALI 'X'            Required, constant 'X'
```

The security exit can be AMODE 24 or 31, and RMODE 24 or ANY. All data areas and parameter lists passed to it are located in storage below 16M.

Figure F-1 shows how a region loads and identifies a full security exit and how Management Services locates the entry point of the exit's code.

*Figure F-1. Management Services—Full Security Exit Linkage*

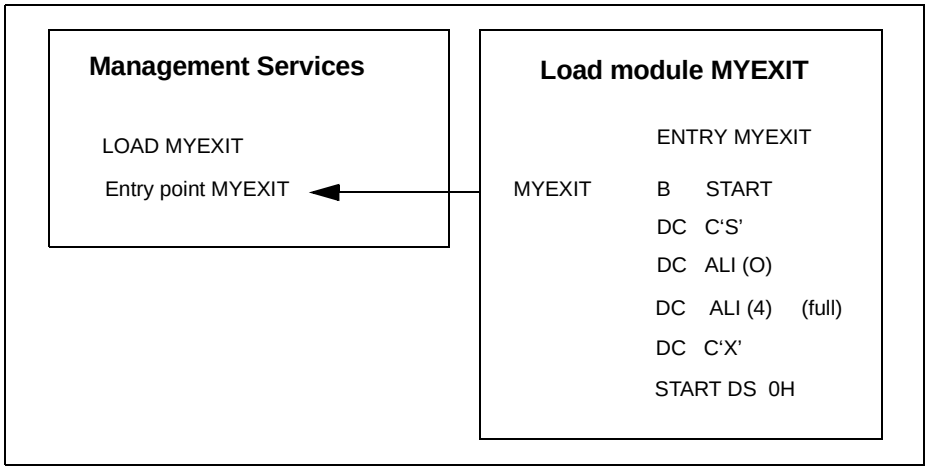
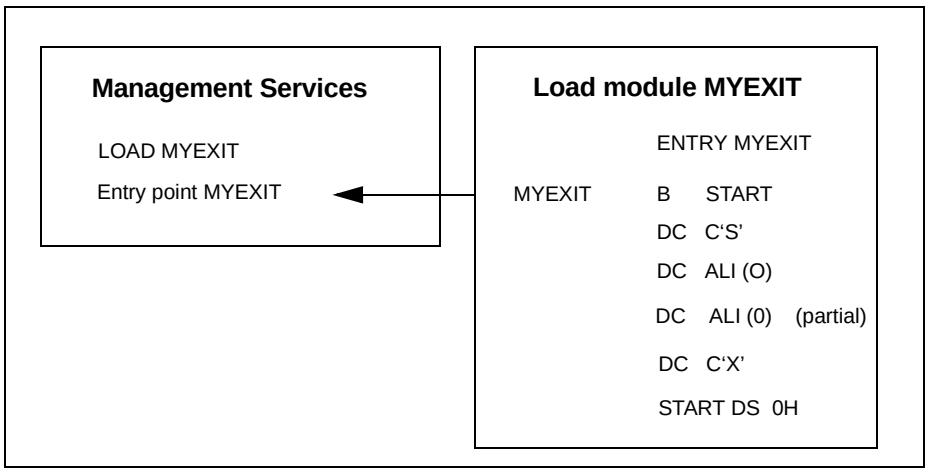



Figure F-2 shows how a region loads and identifies a partial security exit and how Management Services locates the entry point of the exit's code.

*Figure F-2. Management Services—Partial Security Exit Linkage*



## Exit Execution

The exit executes within an operating system subtask and can therefore issue WAITs or SVCs that suspend the task without affecting the Management Services primary task.

**Note**

In OS/VS systems, the subtask is attached sharing subpool 50 with the Management Services main task. This allows other exits (which also use subpool 50) to share common storage areas with the security exit.

Calls to the exit are serialized across the system—if one call to the exit is in progress, any subsequent calls are queued and processed one-by-one, in request order. Serial request processing means that the exit does not have to be written as reentrant.

The exit is called for two reasons:

- Information—to keep the security system informed of developments or system changes
- Function request—to request that a function be performed

**Warning**

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 with the internal abend 268-01.

## Supported Exit Calls

The following table lists the calls supported by the exit, their type, and their function codes:

*Table F-1. Supported Exit Calls*

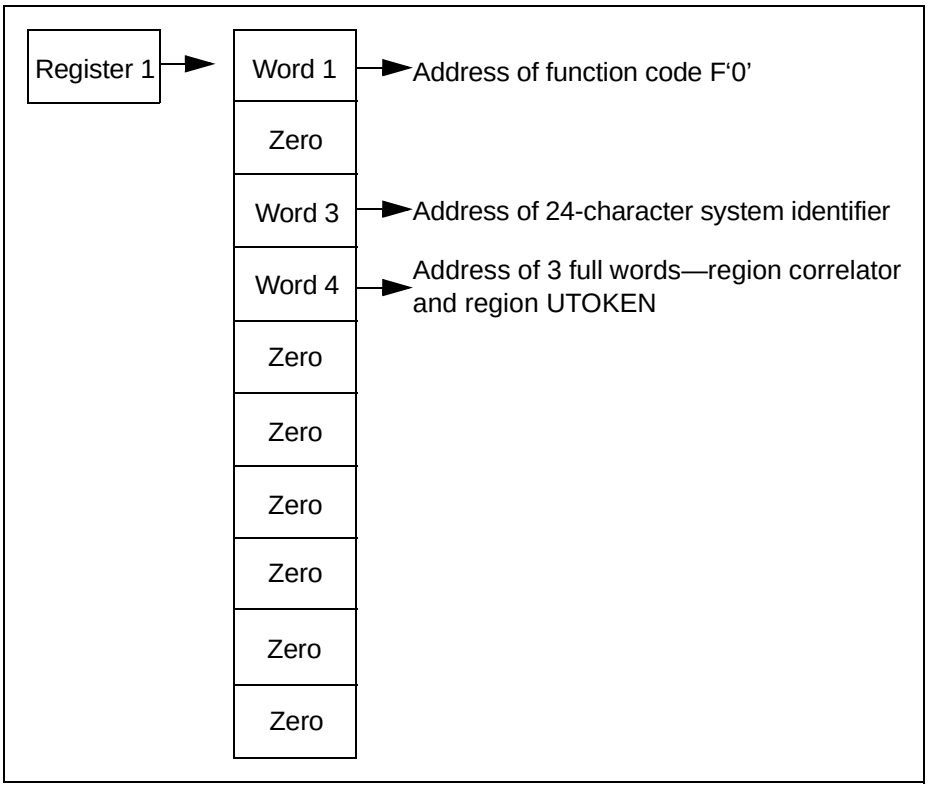
| Call | Type | Function Code |
|---|---|---|
| System Initialization | information | 0 |
| System Close Down | information | 4 |
| User Logon Request | function request | 8 |
| User Logoff Request | information | 12 |
| Logon Verification | function request | 16 |
| Change Password | function request | 20 |
| Return User ID Information | function request | 24 |
| Update User ID | function request | 28 |
| &SECCALL EXIT | function request | 32 |
| Return Sequential User ID Information | function request | 36 |
| Add User ID | function request | 40 |
| Delete User ID | function request | 44 |

The following sections provide the parameter lists for each call.

## The System Initialization Parameter List

On entry to the security exit, Register 1 points to a list of ten full words, as shown in Figure F-3.

*Figure F-3. System Initialization Parameter List*




The contents of the parameter list are as follows:

**Word 1**
A full word containing the address of function code F‘0’.

**Word 2**
Set to zero.

**Word 3**
A 24-character area containing the following:

- The 8-character primary ACB name field (padded to the right with blanks if necessary). The primary ACB name is as specified on the PRI JCL initialization parameter.
- 4 blanks

- The 4-character system domain ID as coded on the NMDID initialization parameter. If the NMDID parameter is omitted the domain ID defaults to the first 4-characters of the primary ACB name used by Management Services (blank padded if required).
- The 4-character system user prefix as coded on the NMSUP initialization parameter. It defaults to the NMDID value if not coded.
- The 4-character Management Services version, for example, V5.0.

**Note**

For further information, see the descriptions of the NMDID and NMSUP parameters on page B-5.

These characters are passed to the exit to identify the system that is executing and might be regarded as the user ID of the system. The exit might want to know this to determine whether, for example, it is a production system.

**Word 4**

The address of three full words. These are as follows:

- First word—can contain any value. This value is returned in the *system closedown call*, so it can be used as a region correlator. The exit can use this to relate this region to a set of control blocks or other information maintained by the exit in relation to the region.
- Second word—the region user token (UTOKEN), if one is available

**Words 5 to 10**

Set to zeros.

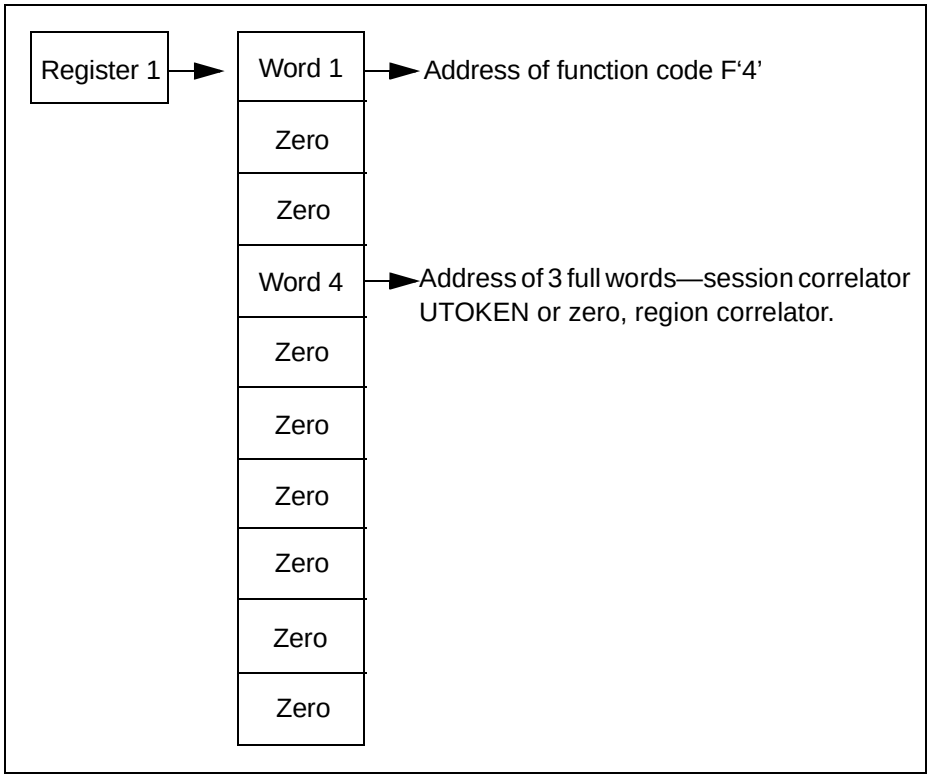
## Return Codes from Initialization Call

The exit must return a completion code in Register 15 on return to Management Services. Successful initialization must be signified by return code F‘0’. Management Services terminates if any other value is returned in Register 15 on completion of this call.

## The System Close Down Parameter List

On entry to the security exit, Register 1 points to a list of ten full words, as shown in Figure F-4.

*Figure F-4. System Close Down Parameter List*




The contents of the parameter list as follows:

**Word 1**
A full word containing the address of function code F‘4’.

**Words 2 and 3**
Set to zeros.

**Word 4**
The address of three full words, each containing the information that was passed on the system initialization call.

**Word 5 to 10**
Set to zeros.

The processing associated with this call depends upon the environment maintained by the exit, but might include closing any datasets used by the exit.

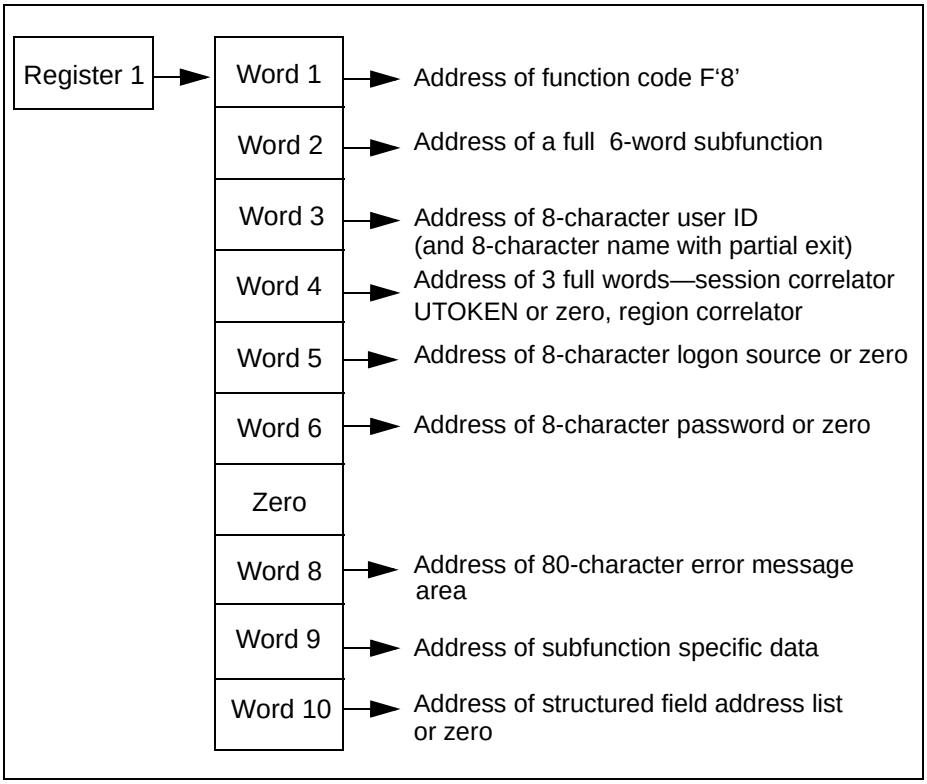
### Return Codes from Closedown Call

The exit must return a completion code in Register 15 on return to Management Services. Successful processing of the close down call must be signified by return code F‘0’.

## The Logon Request Parameter List

On entry to the security exit, Register 1 points to a list of ten full words, as shown in Figure F-5.

*Figure F-5. Logon Request Parameter List*




The contents of the parameter list are as follows:

**Word 1**

A full word containing the address of function code F‘8’.

**Word 2**

Address of a subfunction code. This code defines the type of logon request being processed and has values set as follows:

| | | |
|---|---|---|
| F‘0’ | : | Logon from native terminal |
| F‘4’ | : | Logon from TSO interface component of EIP |
| F‘8’ | : | Logon from ROF user |
| F‘12’ | : | System console logon |
| F‘16’ | : | System environment logon |
| F‘20’ | : | Reserved |
| F‘24’ | : | Reserved |
| F‘28’ | : | APPC user region logon |
| F‘32’ | : | Reserved |

**Word 3**

The address of an 8-character field, blank padded, containing the user ID requesting the logon. If a partial security exit is installed then there is an additional, contiguous 8-character field, blank-padded containing the model user ID specified by the SYSPARMS MODLUSER operand. If no model user has been specified this field is set to blanks.

**Word 4**

The address of three full words. These words are as follows:

- First word—can contain any value to relate a user to a set of control blocks or other information maintained by the exit in relation to that user
- Second word—the address of the user token (UTOKEN) obtained from the SAF logon call is placed in the second full word

These values are returned in all subsequent calls to the exit in reference to this user ID session, so that it can be used as a session correlator.

**Word 5**

Varies according to subfunction code:

| | |
|---|---|
| Subfunction F‘0’ | Word 5 contains the address of an 8-character field, blank-padded, which contains the name of the terminal at which the user wants to logon. |
| Subfunction F‘4’ | Word 5 contains the address of an 8-character field, blank-padded, which contains the name of the VTAM APPL name used by the TSO interface for this session. |
| Subfunction F‘8’ | Word 5 is set to zero. |

| | |
|---|---|
| Subfunction F‘12’ | Word 5 contains the address of an 8-character field holding the word CONSOLE, or the generated user ID for multiple console support (OS/VS systems only). |
| Subfunction F‘16’ | Word 5 contains the address of an 8-character field containing the name of the pseudo-terminal with which the system environment logon is associated. |
| Subfunction F‘28’ | Word 5 is set to zero. |

**Word 6**

Varies according to subfunction code:

| | |
|---|---|
| Subfunction F‘0’ | Word 6 contains the address of an 8-character field, blank-padded, containing the password entered with the user ID. |
| Subfunction F‘4’ | Word 6 is set to zero. |
| Subfunction F‘8’ | Word 6 contains the address of an 8-character field, blank-padded, holding the password entered on a SIGNON command from the remote region. If no password was entered, this word is zero. |
| Subfunction F‘12’ | Word 6 is set to zero. |
| Subfunction F‘16’ | Word 6 is set to zero. |
| Subfunction F‘28’ | Word 6 contains the address of an 8-character field, blank-padded, holding the password specified on the APPC NCL verb that started this region. If no password was entered, this word is zero. |

**Word 7**

Set to zero.

**Word 8**

Address of an 82-character area into which the exit can place any error message text if the logon request is denied. Any text placed in this area is displayed to the user in response to the logon attempt. If an error message is returned in this area it must be formatted as follows:

| | |
|---|---|
| Bytes 00 to 01 | Hex length of message text (excluding these two bytes). |
| Bytes 02 to *nn* | Maximum of 80 characters of error message text. |

Error message text is converted to upper case.

**Word 9**

Varies according to subfunction code.

| | |
|---|---|
| Subfunction F‘0’ | Word 9 set to zero. |
| Subfunction F‘4’ | Word 9 set to zero. |
| Subfunction F‘8’ | Address of 12-character area holding the INMC link name of the system from which a ROF logon request has been received. This is followed by a 4-character field which is either zero or contains the domain ID of the system from which the SIGNON request originated. Following this is a 1-byte binary field with flag settings as follows:<br><br>X‘01’ A character domain ID is present after the 12-character linkname.<br><br>X‘02’ Is set if the system represented by domain ID is different from the system represented by the link name. |
| Subfunction F‘12’ | Word 9 set to zero. |
| Subfunction F‘16’ | Word 9 set to zero. |
| Subfunction F‘28’ | Word 9 set to zero. |

**Word 10**

This word depends on whether you have a full or partial security system:

- Full security exit—the logon is accepted and the exit must provide the address of a list of full words in this field, terminated by X‘FFFFFFFF’.

  Each word in this list contains the address of a structured field defining an item of information for this user ID. Management Services processes all these structured fields to determine the privileges and other information to be associated with the user ID.

- Partial security exit—set to zero, or the address of a variable length list of full words, ended by a full word of X‘FFFFFFFF’.

  Each word in the area points to a single structured field, representing one attribute of the current UAMS definition for the user ID pointed to by word 3 of the parameter list.

If the requesting user ID is not defined on UAMS, Word 10 is zero.

## Return Codes from Logon Call

The exit must return a completion code in Register 15 on return to Management Services. Completion codes are supported as described below. Specific causes for logon rejection are identified by error message text returned in the area addressed by word 8 of the Logon Call parameter list.

The return codes listed below are the only codes that are accepted by Management Services in response to a logon call. Any other return code is treated as indicating that the logon is rejected.

| | |
|---|---|
| 0 | Logon accepted without error. |
| 4 | Logon accepted but password has expired. Management Services is to enforce password renewal before the user has access to any other functions. |
| 8 | Logon accepted but this is a new user ID. Management Services is to enforce password change before allowing the user access to any other functions. |
| 12 | Reserved. Cannot be used as a return code. |
| 16 | Logon rejected. Password is incorrect, but Management Services is to allow a retry. |
| 20 | Logon rejected. Password is incorrect and no retry is to be permitted. Management Services logs a violation message. |
| 24 | Logon rejected. An error message explaining the cause of rejection is available in the 82-character area addressed by word 8 of the parameter list. The format of this area must be a 2-byte length field set to contain the length of the error message text, followed by up to 80 characters of error message text. If the logon was from a TSO user, the error message is not displayed; the normal Management Services logon panel is presented for entry of user ID and password and the user is not classified as a TSO interface user. |

**Note**

Return codes 16 and 20 are provided to give compatibility with the standard UAMS functions, which provide a maximum number of password retries before rejecting the logon attempt and logging a violation message to the Management Services log.

Return codes 4 and 8 provide compatibility with the UAMS convention of enforcing password change at logon time if the user's password has expired, or for the first logon of a new user ID.

The exit might not be able to determine that a user ID is new, depending upon the information available to it from the external security system in use.

If required, the exit can be written to provide two return codes only, 0 or 24, with an appropriate error message if the logon is rejected.

The ability to deny a logon with return code 24 and an error message of the exit's choice allows the installation to extend the security exit's function beyond that of simple password validation. For example, the exit might deny logons after a certain time of day or reject ROF logons from certain remote systems.

System console and system environment logon calls take default attributes if the exit causes the logon to fail.

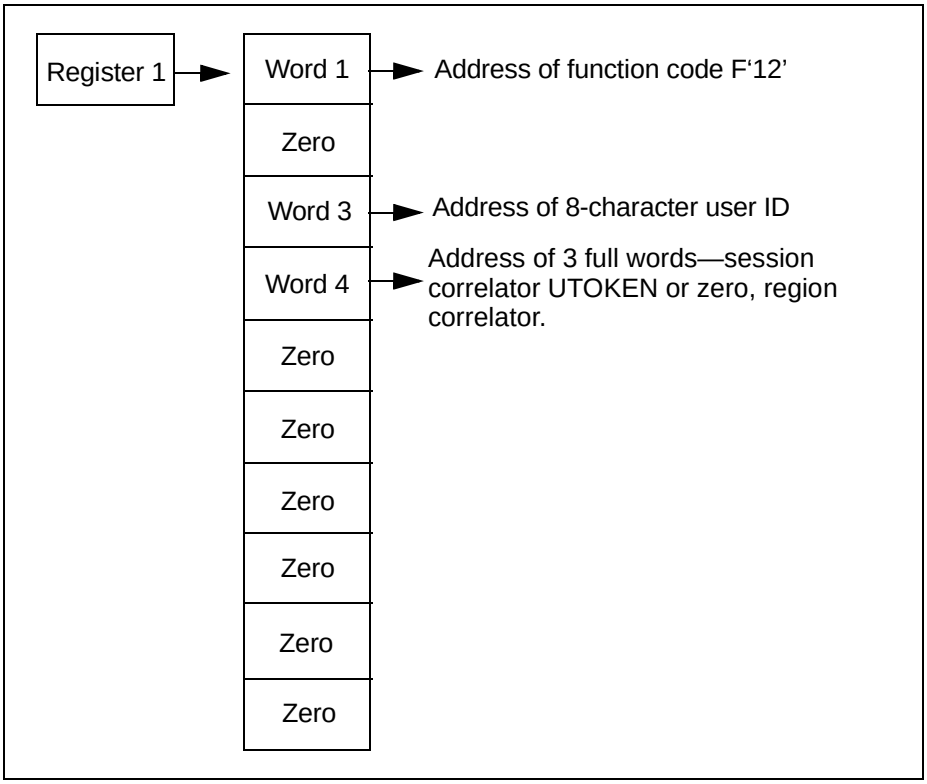
**Note**

See the notes in the section, *Controlling Access to Your System*, on page 4-3, for additional information about this request.

## The Logoff Request Parameter List

On entering the security exit, Register 1 points to a list of ten full words, as shown in Figure F-6.

*Figure F-6. User Logoff Parameter List*




The contents of the parameter list are as follows:

**Word 1**
A full word containing the address of function code F‘12’.

**Word 2**
Set to zero.

**Word 3**
Address of 8-character field, blank-padded, containing the user ID logging off.

**Word 4**
Address of three full words containing the user session correlator (if any) and the user token (UTOKEN) address, set by the exit when this user ID logged on, or zero.

**Words 5 to 10**
Set to zero.

## Return Codes from Logoff Calls

The exit must return a completion code in register 15 on return to Management Services. The only valid completion code is F‘0’.

**Note**

It is possible for the exit to receive a logoff call for a user who has never successfully logged on. This might occur when a user abandons their logon attempt due to a forgotten password. The exit must be written to accept such calls and to ignore them.

Management Services generates a logoff call during DISCONNECT processing on behalf of an internally created environment. As there was no logon request call for this environment, there is no session correlator created, and it is, therefore, null.

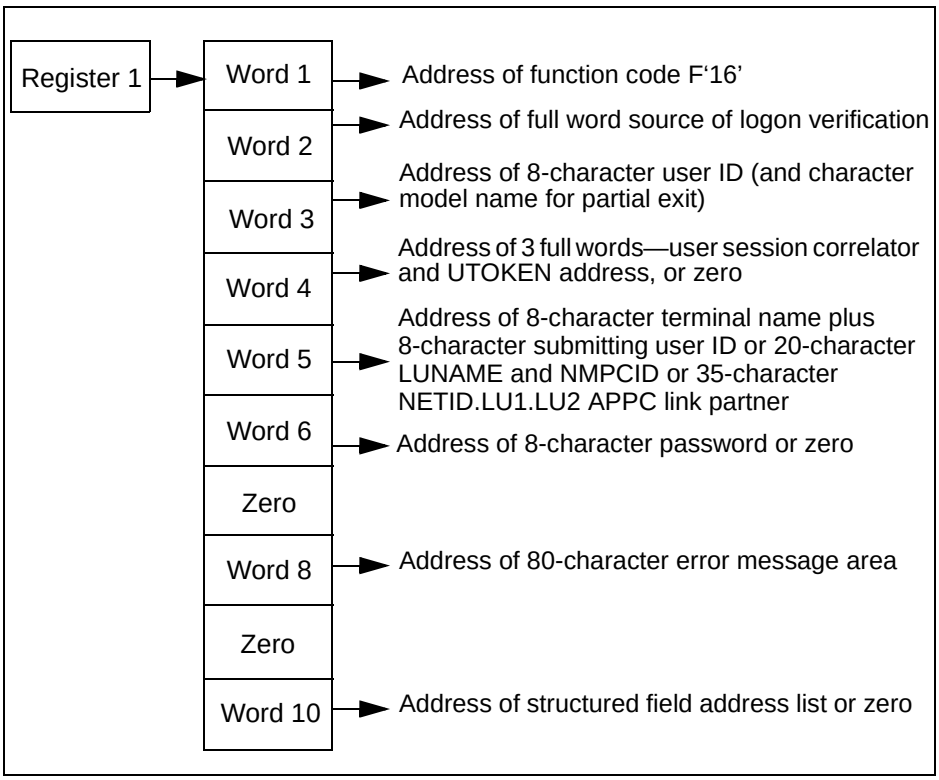
When a users logs back on to reconnect, a logon request call is made to the security exit. A new session correlator is created at this time. When the user selects a RECONNECT or CANCEL of all disconnected environments, a logoff request call is made to the security exit with the original user session correlator. The reconnected or new session continues with a new user session correlator that is presented to the security exit upon that session’s logoff.

A zero user session correlator on a logoff request call to the security exit is valid for a DISCONNECT request only.

## The Logon Verification Call Parameter List

On entry to the security exit, Register 1 points to a list of ten full words, as shown in Figure F-7.

*Figure F-7. Logon Verification Parameter List*




The contents of the parameter list are as follows:

**Word 1**

A full word containing the address of function code F‘16’.

**Word 2**

The address of a subfunction code. This code defines the source of the logon verification call and has values set as follows:

F‘0’ Call is from a &SECCALL CHECK verb, the LOCK facility, or APPC region validation.

F‘4’ Call is from INMC link activation.

F‘8’ Call is from APPC link activation.

**Word 3**

The address of a 16-character field, blank-padded, containing the 8-character-character user ID whose password and logon capability is to be verified, followed by the 8-character system default MODLUSER name, or blanks if no model user has been defined.

**Word 4**

The address of three full words containing the user session correlator and the address of the user token (UTOKEN) of the submitter of the logon verification call, or zero if called from EASINET.

**Word 5**

Varies according to subfunction code:

| | |
|---|---|
| Subfunction F‘0’ | Address of the 8-character terminal ID, followed by the 8-character user ID of the submitting user. This field can be blank; if so, treat the user ID being checked as the submitting user. |
| Subfunction F‘4’ | Address of the 8-character LU name of the link, followed by the 12-character NMPCID. |
| Subfunction F‘8’ | Address of the 35-character APPC link partners. The format of the link session partners is NETID.LU1.LU2 where NETID.LU1 is the source partner and LU2 is the destination LU of the APPC link. |

**Word 6**

Varies according to subfunction code:

| | |
|---|---|
| Subfunction F‘0’ | Address of the 8-character, blank-padded user ID password. |
| Subfunction F‘4’ | Word 6 is set to zero. |
| Subfunction F‘8’ | Address of an 8-character area into which the exit can place the session key for this APPC link. |

**Word 7**

Set to zero.

**Word 8**

Address of an 82-character area into which the exit can place an error message. If an error message is returned, it must be formatted as follows:

| | |
|---|---|
| Bytes 00 to 01 | Hex length of message text (excluding these two bytes). |
| Bytes 02 to *nn* | Maximum of 80 characters of error message text. |

Error message text is converted to upper case. An error message might be returned in this area for return codes 20 or 24. A message set in this area for any other return code is ignored.

**Word 9**

Set to zero.

**Word 10**

If you have a full security exit, this word is set to zero.

If you have a partial security exit this word is set to zero if the user ID is not defined on the UAMS dataset. Otherwise word 10 points to a list of full words, ended by a full word of X'FFFFFFFF'. Each address in the list points to a single structured field representing one attribute of the user ID to be verified. This list therefore provides the exit with access to the entire current definition of this user ID.

## Return Codes from Logon Verification Call

The exit must return a completion code in Register 15 on return to Management Services. Completion codes are supported as described below.

The return codes listed below are for subfunction codes 0 and 4. They are the only codes that are accepted by Management Services in response to a Logon Verification call. Any other return code is rejected and set to 24.

| | |
|---|---|
| 0 | Password is valid. Logon is successful. |
| 4 | Password is valid but expired. This is not a new user ID. The user is prompted to change the password before logon is successful. |
| 8 | Password is valid but this is a new user ID. The user is prompted to change the password before logon is successful. |
| 16 | Password is wrong. |
| 20 | Password is correct but logon is rejected. The exit might return an explanatory message in the error message area addressed by word 8 of the parameter list. |
| 24 | Request failed or function is not supported by the exit. The exit might return an explanatory message in the error message area addressed by word 8 of the parameter list. |

The return codes listed below are for subfunction code 8. They are the only codes that are accepted by Management Services in response to an APPC link establishment Logon Verification call. Any other return code is rejected and set to 24.

| | |
|---|---|
| 0 | Link activation successful. The session key is addressed by word 6 of the parameter list. |
| 4 | Link activation successful. No session key is available. |
| 8 | N/A |
| 16 | Link activation is unsuccessful. |
| 20 | N/A |
| 24 | Function not supported. The link establishment is to continue without a session key. |

**Note**

When a Logon Verification Call to the exit is made by a &SECCALL CHECK statement or by NMPC link activation, no user correlator information can be provided. Word 4 of the parameter list is set to zero.

The Logon Verification Call return codes do not strictly correlate with the &SECCALL CHECK return code values. See the *Network Control Language Reference,* for an explanation of &SECCALL CHECK return codes.

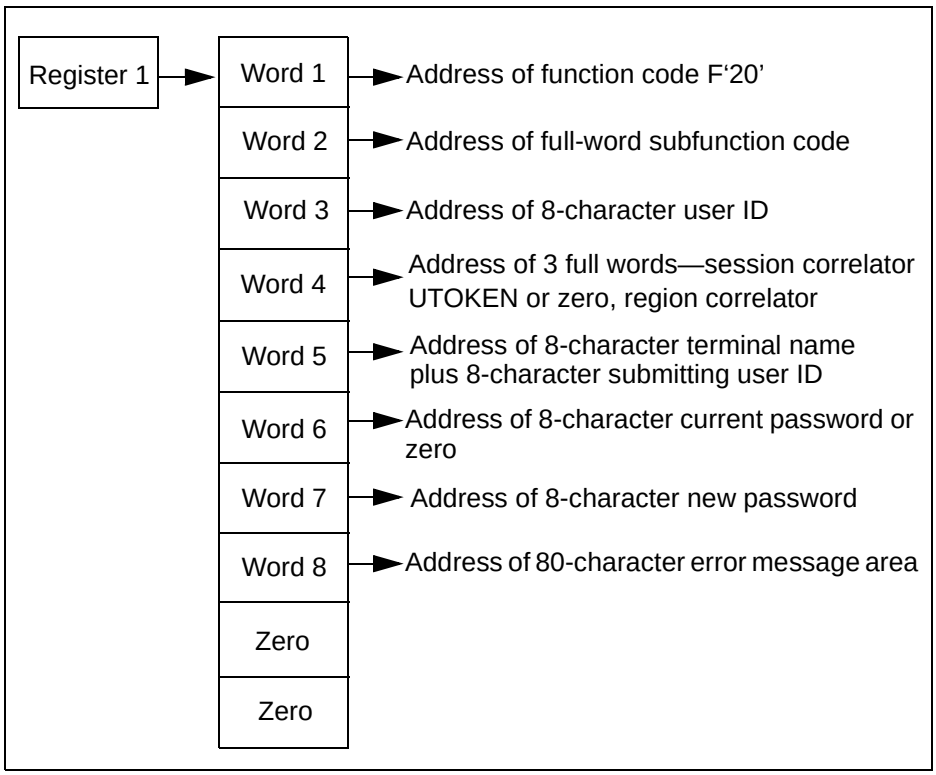
If the Logon Verification call is made by the LOCK facility, the call is being made on behalf of a logged-on user. Consequently, the parameter list provides the address of the appropriate user session correlator.

The Logon Verification Call is primarily a means of querying the validity of a password, and the exit can be written to provide only this function. The option of supporting return code 20 is provided to complement support of return code 24 from the Logon Request Call, which allows the exit to refuse a logon request for reasons other than password error.

## The Change Password Parameter List

On entry to the security exit, Register 1 points to a list of ten full words, as shown in Figure F-8.

*Figure F-8. Change Password Parameter List*




The contents of the parameter list are as follows:

**Word 1**

A full word containing the address of function code F‘20’.

**Word 2**

Address of a subfunction code. This code defines the source of the change password request and has values set as follows:

F‘0’ User has requested password change

F‘4’ Change required because password expired

F‘8’ Forced change by UAMS user

F‘12’ &SECCALL CHANGE request issued from an EASINET procedure

**Word 3**

The address of an 8-character field, blank-padded, containing the user ID whose password is to be changed.

**Word 4**

Varies according to subfunction code:

| | |
|---|---|
| Subfunction F‘0’ | Word 4 is the address of two full words. The first contains the correlator and the second the user token (UTOKEN) of the user issuing the change request. |
| Subfunction F‘4’ | As for subfunction F‘0’. |
| Subfunction F‘8’ | As for subfunction F‘0’. |
| Subfunction F‘12’ | Word 4 is zero. |

**Word 5**

The address of 8-character terminal name followed by the 8-character user ID of the user issuing the change request.

**Word 6**

Varies according to subfunction code:

| | |
|---|---|
| Subfunction F‘0’ | Word 6 contains the address of an 8-character field, blank-padded, which contains the current user ID password. |
| Subfunction F‘4’ | As for subfunction F‘0’. |
| Subfunction F‘8’ | Word 6 is set to zero. |
| Subfunction F‘12’ | As for subfunction F‘0’. |

**Word 7**

The address of an 8-character field containing the new password to be assigned to the user ID.

**Word 8**

The address of an 82-character area into which the exit can place any error message text if the change request is denied or failed. Any text placed in this area is displayed to the user in response to the change request. If an error message is returned in this area it must be formatted as follows:

| | |
|---|---|
| Bytes 00 to 01 | Hex length of message text (excluding these two bytes). |
| Bytes 02 to *nn* | Maximum of 80 characters of error message text. |

**Words 9 to 10**
Set to zero.

## Return Codes from Change Call

The exit must return a completion code in Register 15 on return to Management Services. The supported completion codes are described below. Specific causes for change rejection are identified by error message text returned in the area addressed by word 8 of the Change Call parameter list.

The return codes listed below are the only codes that are accepted by Management Services in response to a Change Call. Any other return code is regarded as indicating that the password was not changed.

| | |
|---|---|
| 0 | Password has been changed. |
| 4 | Subfunction is not supported by exit or other error. A message explaining the cause of error might be placed in the 80-character area addressed by word 8 of the parameter list. |
| 8 | Password has not been changed. A message explaining the cause of rejection is available in the 80-character area addressed by word 8 of the parameter list. |

**Note**

Management Services does not apply minimum password length checks, nor is automatic password expiry provided. The exit is responsible for all maintenance and knowledge of passwords.

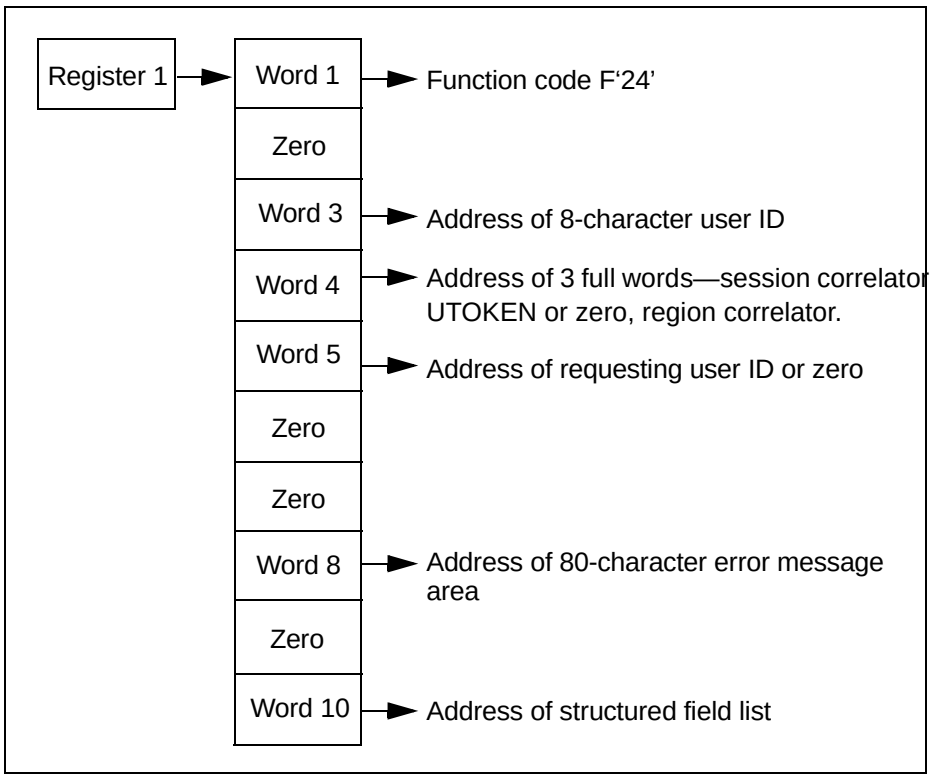
A user's current password is provided as a parameter for the change call only for subfunctions 0, 4, and 12. That is, when the change request is made by the user after logging on, when the user is required to change the password because it has expired or by EASINET using the &SECCALL CHECK statement.

For subfunction 8, when a UAMS privileged user requests a forced password change for another user ID it is almost certainly because the current password for that user ID has been forgotten, and is therefore not available for presentation as a parameter for this call.

## The Return User ID Information Parameter List

On entry to the security exit, Register 1 points to a list of ten full words, as shown in Figure F-9.

*Figure F-9. Return User ID Information Parameter List*




The contents of the parameter list are as follows:

**Word 1**
A full word containing the address of function code F‘24’.

**Word 2**
Set to zero.

**Word 3**
The address of an 8-character field, blank-padded, containing the user ID for which information is required.

**Word 4**
The address of three full words containing the session correlator and the user token (UTOKEN) associated with the user issuing the request (not necessarily the correlator for the user ID whose information is being requested).

**Word 5**

Address of an 8-character field, blank-padded, containing the user ID requesting the information (or zero).

**Word 6**

Set to zero.

**Word 7**

Set to zero.

**Word 8**

Address of an 82-character area into which the exit can place any error message text if the function request is denied or failed. Any text placed in this area is displayed to the user in response to the command that caused this request. If an error message is returned it must be formatted as follows:

| | |
|---|---|
| Bytes 00 to 01 | Hex length of message text (excluding these two bytes). |
| Bytes 02 to *nn* | Maximum 80 characters of error message text. |

**Word 9**

Set to zero.

**Word 10**

This word is set differently depending on the type of exit you have:

- Full security exit—if the request is honored, the exit must place in this word the address of a list of full words, terminated by X‘FFFFFFFF’.

  Each full word in this list in turn points to a structured field that defines an item of information relevant to the user ID.

- Partial security exit—the address of a list of full words, terminated by X‘FFFFFFFF’.

  Each full word in this list in turn points to a structured field that defines an item of information relevant to the user ID.

The format and contents of the structured fields are described in Appendix G, *Structured Fields*.

## Return Codes from Return User ID Information Call

The exit must return a completion code in Register 15 on return to Management Services. Completion codes are supported as described below. Causes for rejection are identified by error message text returned in the area addressed by word 8 of the Information Call parameter list.

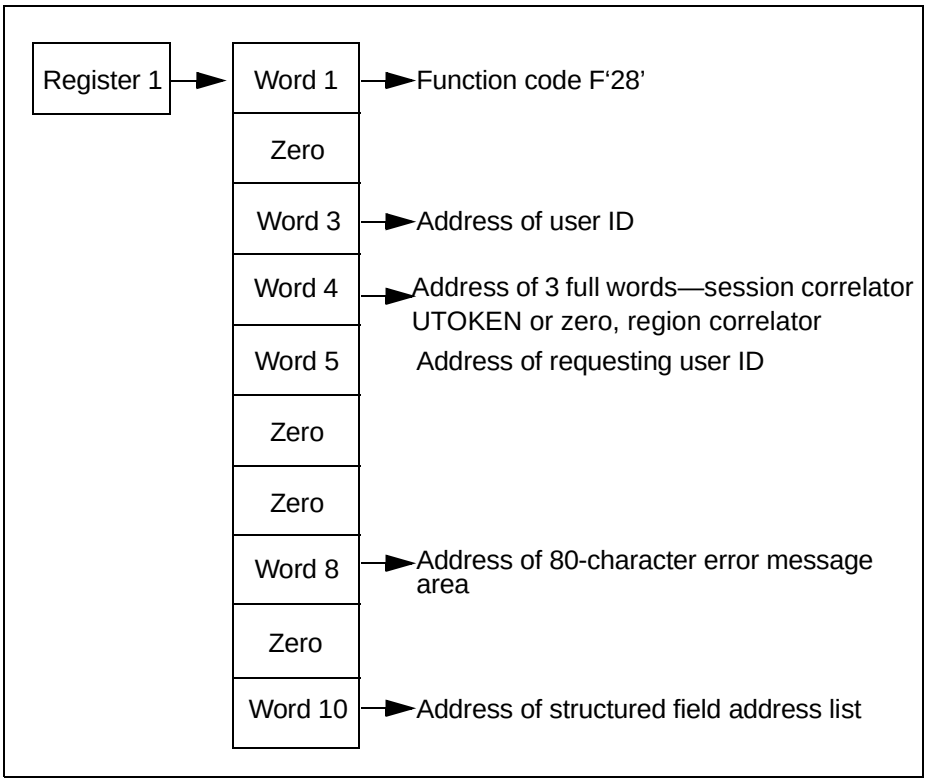
The return codes listed below are the only codes that are accepted by Management Services in response to an Information Call. Any other return code is treated as information not available.

| | |
|---|---|
| 0 | Information available. Word 10 of the parameter list points to an address list that provides pointers to structured fields describing the user ID. |
| 4 | Information not available. An error message should be available in the 80-character area addressed by word 8 of the parameter list. |

## The Update User ID Parameter List

On entry to the security exit, Register 1 points to a list of ten full words, as shown in Figure F-10.

*Figure F-10. Update User ID Parameter List*



The contents of the parameter list are as follows:

**Word 1**
A full word containing the address of function code F‘28’.

**Word 2**
Set to zero.

**Word 3**
The address of an 8-character field, blank-padded, containing the name of the user ID to be updated.

**Word 4**
The address of three full words containing the session correlator and the user token (UTOKEN) associated with the user issuing the request.

**Word 5**
The address of an 8-character field containing the user ID of the user making the update request.

**Word 6**
Set to zero.

**Word 7**
Set to zero.

**Word 8**
The address of an 82-character area into which the exit can place any error message text if the function request is denied or failed. Any text placed in this area is displayed to the user in response to the command that caused this request. If an error message is returned in this area it must be formatted as follows:

| | |
|---|---|
| Bytes 00 to 01 | Hex length of message text (excluding these two bytes). |
| Bytes 02 to *nn* | Maximum of 80 characters of error message text. Error message text is converted to upper case. |

**Word 9**
Set to zero.

**Word 10**
Set to the address of a list of full words, terminated by X‘FFFFFFFF’. Each full word in this list in turn points to a structured field that defines an item of user ID information that is to be changed.

The format and contents of the structured fields are described in Appendix G, *Structured Fields*.

### Return Codes from Update User ID Information Call

The exit must return a completion code in Register 15 on return to Management Services. Completion codes are supported as described below. Causes for update rejection are identified by error message text returned in the area addressed by word 8 of the Update Call parameter list.

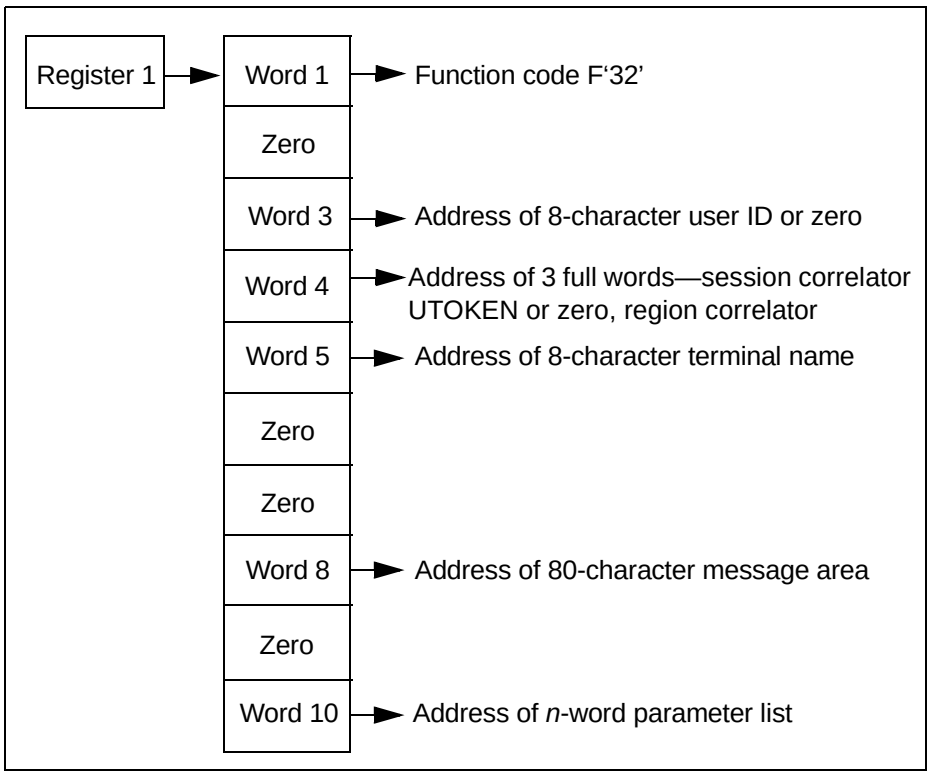
The return codes listed below are the only codes that are accepted by Management Services in response to an Update Call. Any other return code is regarded as update rejected.

| | |
|---|---|
| 0 | Update completed. |
| 4 | Update rejected. An error message might be available in the 80-character area addressed by word 8 of the parameter list. |

## The &SECCALL EXIT Parameter List

On entry to the security exit, Register 1 points to a list of ten full words, as shown in Figure F-11.

*Figure F-11. &SECCALL EXIT Parameter List*



The contents of the parameter list are as follows:

**Word 1**
A full word containing the address of function code F‘32’.

**Word 2**
Set to zero.

**Word 3**
The address of an 8-character field, blank-padded, containing the user ID executing the procedure in which &SECCALL EXIT was issued. If &SECCALL EXIT was issued from an EASINET procedure, this word is set to zero.

**Word 4**
The address of three full words containing the session correlator and the user token (UTOKEN) associated with the user issuing the &SECCALL EXIT statement. If no user ID is involved because the &SECCALL EXIT statement has been issued from an EASINET procedure, this word is set to zero.

**Word 5**
The address of 8-character terminal name.

**Word 6**
Set to zero.

**Word 7**
Set to zero.

**Word 8**
The address of an 82-character area into which the exit can place any message text. If a message is placed in this area it is returned to the issuing NCL procedure in the system variable &SYSMSG. If a message is returned in this area it must be formatted as follows:

| | |
|---|---|
| Bytes 00 to 01 | Hex length of message text (excluding these two bytes). |
| Bytes 02 to *nn* | Maximum of 80 characters of error message text. Message text is converted to upper case. |

**Word 9**
Set to zero.

**Word 10**

Address of a variable length list of contiguous full words, ended by a full word containing a value of X‘FFFFFFFF’. Each word in the list points at an area that represents an NCL variable, where the format of this area is always:

| | |
|---|---|
| Bytes 00 to 01 | Length of parameter data present (excluding these 2 length bytes). |
| Bytes 02 to 257 | Each parameter is padded to 256 bytes with blanks. |

## Return Codes from &SECCALL EXIT Call

The exit can pass information back to the calling NCL procedure in the following ways:

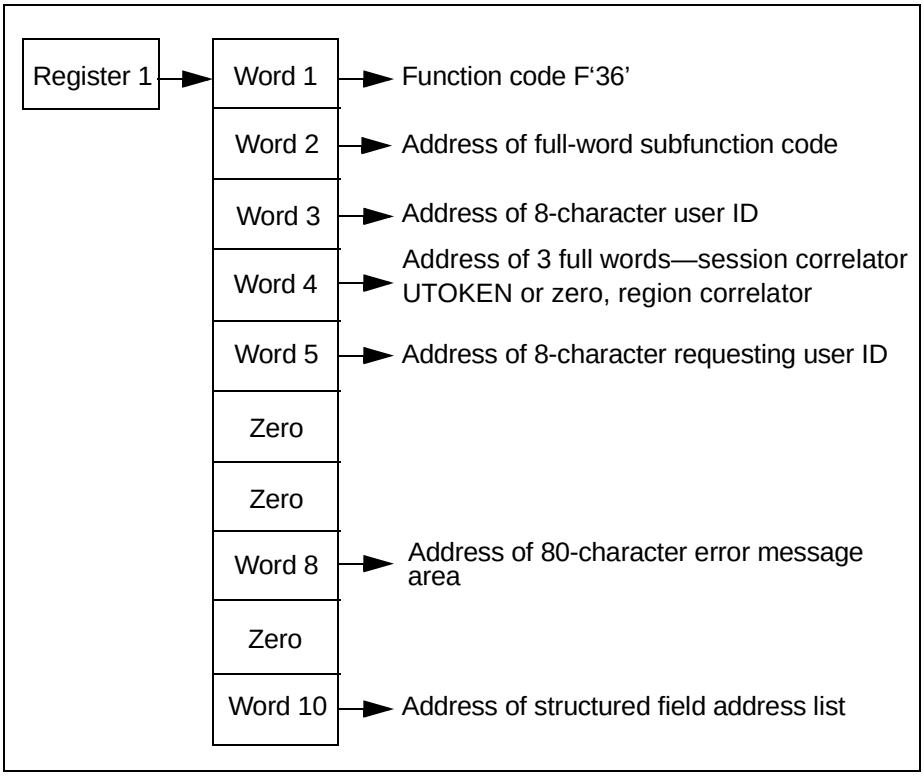
- A return code can be set by the exit in Register 15 on exit. Valid return codes are 0 to 99. Any other return code is forced to 99. The return code is available to the procedure in the system variable &RETCODE after the &SECCALL EXIT statement.
- A message (error or otherwise) can be returned in the 82-character message area addressed by word 8 of the parameter list, as described above.
- Information can be returned in the parameter areas addressed by word 10 of the &SECCALL EXIT parameter list. The exit can return information only in those areas originally passed in the parameter list. Information returned in these areas must be formatted as follows:

| | |
|---|---|
| Bytes 00 to 01 | Length of parameter data present (excluding these 2 length bytes). |
| Bytes 02 to 257 | Parameter data. The maximum length for each return parameter is 256. |

## The Return Sequential User ID Information Parameter List

On entry to the security exit, Register 1 points to a list of ten full words, as shown in Figure F-12.

*Figure F-12. Return SEQUENTIAL User ID Information List*




The contents of the parameter list are as follows:

**Word 1**

A full word containing the address of function code F‘36’.

**Word 2**

The address of a subfunction code. The code defines the type of sequential call being requested and has values set as follows:

F‘0’ Retrieve the next user definition following the key user ID (provided in word 3). The user ID for which information is being requested is the next user ID in the collating sequence.

F‘4’ Retrieve the previous user definition preceding the key user ID (provided in word 3). The user ID for which information is being requested is the previous user ID in the collating sequence.

**Word 3**

The address of an 8-character field, blank-padded, containing the key user ID for the call, or set to zero if the request is issued for the first (next), or last (previous) user ID in the collating sequence.

**Word 4**

The address of three full words containing the session correlator and the user token associated with the user issuing the request (not necessarily the correlator for the user ID whose information is being requested).

**Word 5**

The address of an 8-character field containing the requesting user ID.

**Word 6**

Set to zero.

**Word 7**

Set to zero.

**Word 8**

Address of an 82-character area into which the exit can place any error message text if the function request is denied or failed. Any text placed in this area is returned to the issuing procedure in the &SYSMSG system variable. If an error message is returned it must be formatted as follows:

| | |
|---|---|
| Bytes 00 to 01 | Hex length of message text (excluding these two bytes). |
| Bytes 02 to *nn* | Maximum 80 characters of error message text. Error message text is converted to upper case. |

**Word 9**

Set to zero.

**Word 10**

If the request is honored, the exit must place in this word the address of a list of full words, terminated by X‘FFFFFFFF’. Each full word in this list in turn points to a structured field that defines an item of information relevant to the user ID.

The format and contents of the structured fields are described in Appendix G, *Structured Fields*.

## Return Codes from Return Next User ID Information Call

The exit must return a completion code in Register 15 on return to Management Services. Completion codes are supported as described below. Causes for rejection are identified by error message text returned in the area addressed by word 8 of the parameter list.

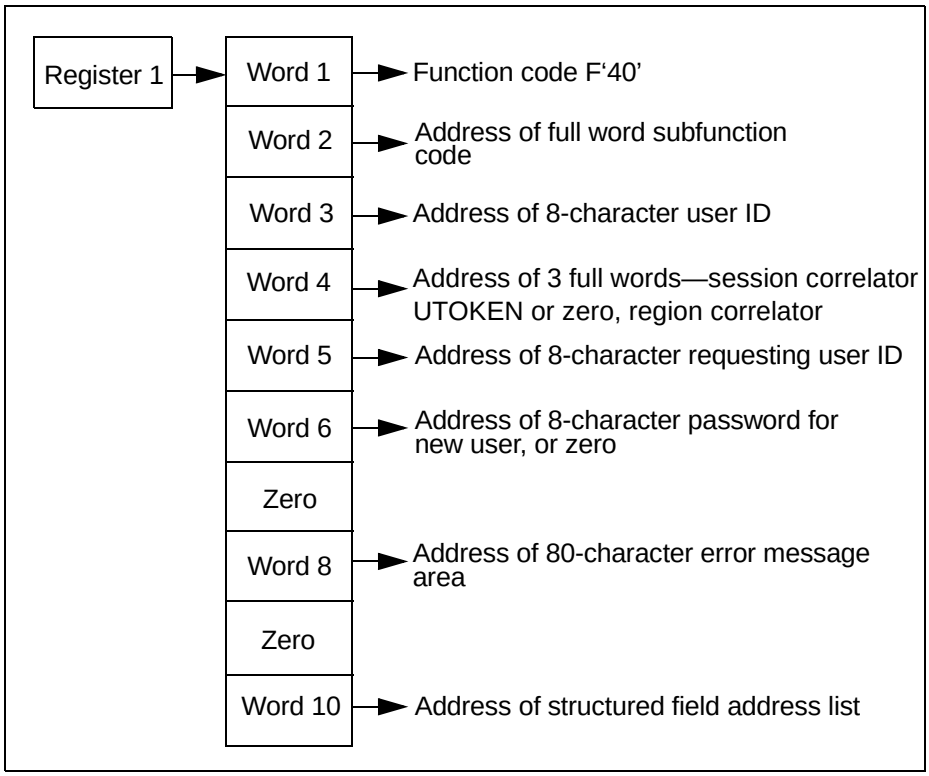
The return codes listed below are the only codes that are accepted by Management Services in response to the call. Any other return code is treated as return code 4.

| | |
|---|---|
| 0 | Information available. Word 10 of the parameter list points to an address list that provides pointers to structured fields describing the user ID. |
| 4 | Information not available. An error message might be available in the 80-character area addressed by word 8 of the parameter list. |

## The Add User ID Parameter List

On entry to the security exit, Register 1 (R1) points to a list of ten full words, as shown in Figure F-13.

*Figure F-13. Add New User ID Parameter List*




The contents of the parameter list are as follows:

**Word 1**

A full word containing the address of function code F‘40’.

**Word 2**

The address of a subfunction code. The code defines the type of user to be added and has values set as follows:

F‘0’ A USER definition is to be added.

F‘4’ A GROUP definition is to be added.

**Word 3**

The address of an 8-character field, padded with blanks, containing the user definition to be added.

**Word 4**

The address of three full words containing the session correlator and the user token (UTOKEN) associated with the user issuing the request.

**Word 5**

The address of a 8-character field containing the user ID of the user issuing the ADD request.

**Word 6**

The address of a 8-character field containing the initial password of the user to be added or set to zero if a password is not provided.

This field is optional on the &SECCALL ADD function when using a security exit. It is the responsibility of the installation to take appropriate action on the requirement for this password.

**Word 7**

Set to zero.

**Word 8**

The address of an 82-character message area into which the exit can place any error message text if the add request is denied or failed. Any text placed in this area is displayed to the user in response to the command that caused this request. If an error message is returned in this area, it must be formatted as follows:

| | |
|---|---|
| Bytes 00 to 01 | Hex length of message text (excluding these 2 bytes). |
| Bytes 02 to *nn* | Maximum of 80 characters of error message text. Error message text is converted to upper case. |

**Word 9**

Set to zero.

**Word 10**

Set to the address of a list of full words, terminated by X'FFFFFFFF'. Each full word in this list in turn points to a structured field that defines an item of user ID information that is to be added.

The format and contents of the structured fields are described in Appendix G, *Structured Fields*.

## Return Codes from the Add User ID Call

The exit must return a completion code in Register 15 (R15) on return to Management Services. Completion codes are supported as described below. Causes for rejection of add calls are identified by error message text returned in the area addressed by word 8 of the add call parameter list.

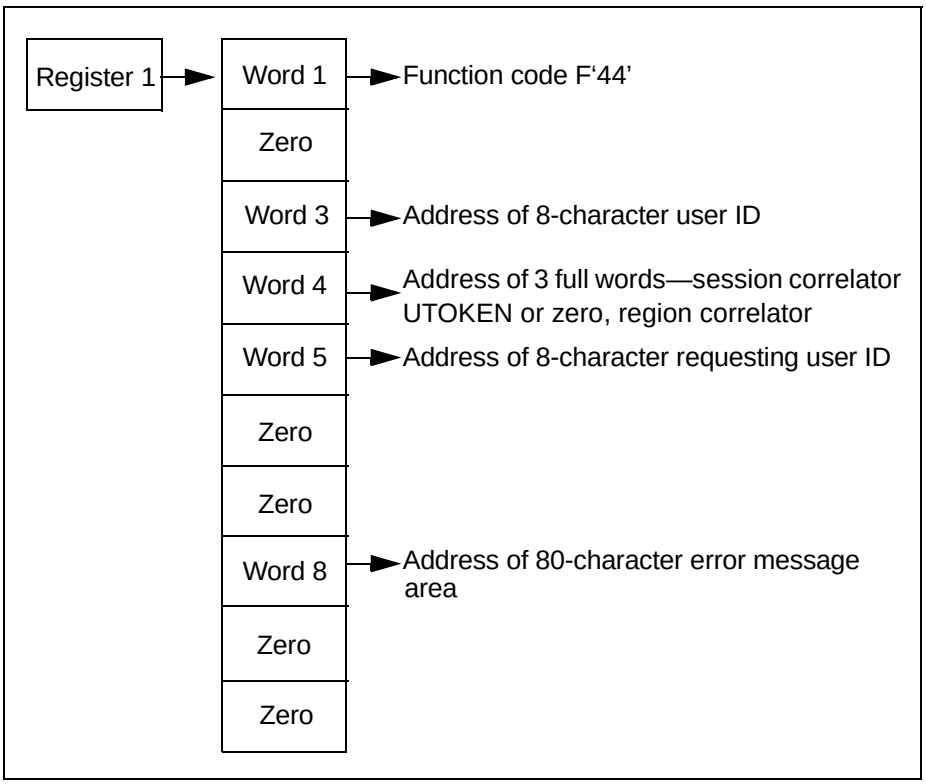
The return codes listed below are the only codes that are accepted by Management Services in response to an add call. Any other return codes are regarded as a rejection of the add call.

| | |
|---|---|
| 0 | ADD complete. |
| 4 | ADD rejected. An error message might be available in the 80-character area addressed by word 8 of the parameter list. |

## Delete User ID Parameter List

On entry to the security exit, Register 1 (R1) points to a list of ten full words, as shown in Figure F-14.

*Figure F-14. Delete User ID Parameter List*




The contents of the parameter list are as follows:

**Word 1**
A full word containing the address of function code F‘44’.

**Word 2**
Set to zero.

**Word 3**
The address of an 8-character field, blank-padded, containing the user definition to be deleted.

**Word 4**
The address of three full words containing the session correlator and the user token (UTOKEN) associated with the user issuing the DELETE request.

**Word 5**

The address of a 8-character field containing the user ID of the user issuing the DELETE request.

**Word 6**

Set to zero.

**Word 7**

Set to zero.

**Word 8**

The address of a 82-character message area into which the exit can place any error message text if the delete request is denied or failed. Any text placed in this area is displayed to the user in response to the command that caused this request. If an error message is returned in this area, it must be formatted as follows:

| | |
|---|---|
| Bytes 00 to 01 | Hex length of message text (excluding these 2 bytes). |
| Bytes 02 to *nn* | Maximum of 80 characters of error message text. Error message text is converted to upper case. |

**Word 9**

Set to zero.

**Word 10**

Set to zero.

## Return Codes from the Delete User ID Call

The exit must return a completion code in Register 15 (R15) on return to Management Services. Completion codes are supported as described below. Causes for delete rejection are identified by error message text returned in the area addressed by word 8 of the DELETE call parameter list.

The return codes listed below are the only codes that are accepted by Management Services in response to an delete call. Any other return codes are regarded as a rejection of the delete call.

| | |
|---|---|
| 0 | Delete complete. |
| 4 | Delete rejected. An error message might be available in the 80-character area addressed by word 8 of the parameter list. |

# G

# Structured Fields

This appendix provides information about structured fields and provides a description of each field available to Management Services.

## About Structured Fields

Each structured field identifies an item of information that can be specified when a user ID is defined. For example, user name, user location, and so on. These structured fields provide the same information as the fields in the Management Services UAMS function for defining a user ID.

For detailed descriptions of the privileges and attributes that a user ID can have, see Chapter 3, *Implementing Security*.

Structured fields are used to exchange information between security exits and Management Services. When an exit returns user ID information to Management Services it is always in the form of a list of pointers to one or more structured fields. By processing the fields returned by the exit, Management Services builds its own internal representation of the user ID.

Management Services also translates its internal representation of a user ID into the same set of structured fields when a request is passed to an exit.

## Format of Structured Fields

The format of a structured field is as follows:

| Byte | Description |
|---|---|
| 00 to 01 | Field key. A hexadecimal value that defines the item of information described by this field. |
| 02 to 03 | Subfield count. A hexadecimal count of the number of subfields within this structured field. |
| 04 to 05 | Subfield length. The hexadecimal length of the following subfield. |
| 06 to *nn* | Subfield data. |

Subfield length and subfield data are repeated according to the number specified in subfield count.

### Example

Figure G-1 is structured field X'0012'. This field is used to define a user's name. This name can be associated with the user ID so that the person can be identified when they logon.

Structured field X'0012' consists of:

- Field key of X'0012'
- Subfield count of X'0001' (1 subfield)
- Subfield length of X'0014' (20 characters)
- Subfield data of JOHN SMITH (padded with blanks on the right to make up the 20 characters).

*Figure G-1. Structured Field Example 1*

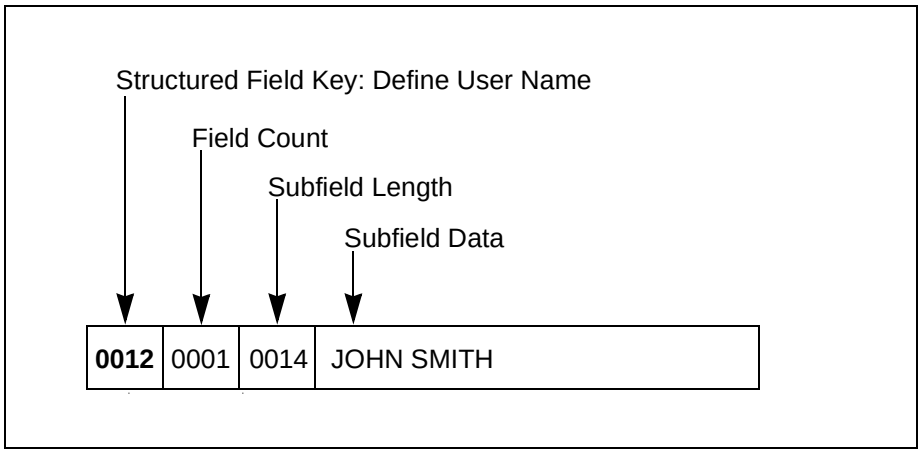
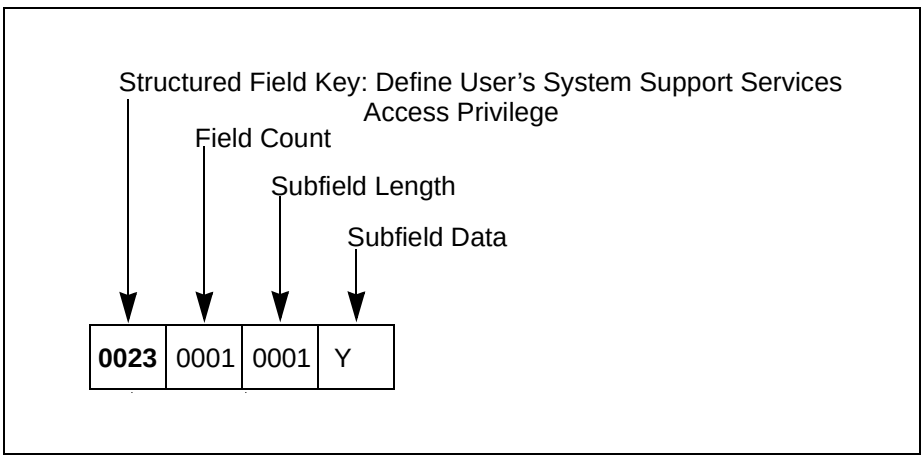



Figure G-2 is structured field X‘0023’. This too has a single subfield with a value of Y and is used to indicate that the user is entitled to access the System Support Services function of Management Services.

*Figure G-2. Structured Field Example 2*




## Updating a Structured Field

For examples of how to update structured fields, including those with multiple subfields, see the description of the &SECCALL verb in the *Network Control Language Reference* manual.

### Structured Field Error Conditions

Error conditions can occur during the processing of a set of structured fields due to one of the following reasons:

- The omission of a structured field
- An invalid value has been specified in a structured field

If an error does occur during structured field processing, Management Services performs the following:

- The structured field is assigned its default value
- An error message is written to the log

If error conditions occurred, a user might not have all the expected privileges when logged on to the system.

### Structured Field Sequences

Management Services processes structured fields in the order in which they appear. While most structured fields are independent of other fields, certain fields need to be processed before others. Where such dependencies exist, they are noted in the description of the fields concerned.

## Structured Field Descriptions

All structured fields that are supported by Management Services are described in the remainder of this appendix. The description of each field includes comments on special processing or validation requirements.

**Note**

Optional products (for example, TCPIP, SNA, SNAACCESS, or FTS) may require other structured fields to be supported in order to assign the appropriate privileges to user IDs. In these cases see the appropriate product manual.

The length of each subfield is fixed and cannot be varied. Where applicable, data fields must be blank padded to the full length of the subfield.

## Structured Field Key X‘0010’—User ID Name

| | |
|---|---|
| FUNCTION: | Define user ID name |
| KEY: | X‘0010’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |

The user ID name is any 8 character string forming a valid user ID. The name must be left justified and blank-padded to a length of 8 characters. This field is optional. If generated by the exit in response to a call for user ID information must be the same user ID name as that requested by Management Services. If omitted, Management Services defaults to the user ID name requested.

## Structured Field Key X‘0011’—User Name

| | |
|---|---|
| FUNCTION: | Define user name |
| KEY: | X‘0011’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0014’ |
| SUBFIELD 1 DATA: | CL20‘ ’ |

The user name is any 20 character string. It allows the name of the user (or any other comments) to be displayed at log on time. This field is optional. If omitted, the name field defaults to blanks. If provided, the subfield must be 20 characters long.

## Structured Field Key X‘0012’—User Location

| | |
|---|---|
| FUNCTION: | Define user location |
| KEY: | X‘0012’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0014’ |
| SUBFIELD 1 DATA: | CL20‘ ’ |

The user location is any 20 character string. This field is optional. If omitted, the location field defaults to blanks. If provided, the subfield must be 20 characters long.

## Structured Field Key X‘0013’—User Telephone Number

| | |
|---|---|
| FUNCTION: | Define user telephone number |
| KEY: | X‘0013’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0014’ |
| SUBFIELD 1 DATA: | CL20‘ ’ |

The user telephone number field is any 20 character string. This field is optional. If omitted, the field will default to blanks. If provided, the subfield must be 20 characters long.

## Structured Field Key X‘0014’—User Language Code

| | |
|---|---|
| FUNCTION: | Define user language code |
| KEY: | X‘0014’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0002’ |
| SUBFIELD 1 DATA: | CL2‘ ’ |

The user language code field is any 2 character string. This field is optional. If omitted, the field will default to blanks. If provided, the subfield must be 2 characters long. If omitted, the field defaults to the current setting of the system language code set by the SYSPARMS LANG operand.

## Structured Field Key X‘0015’—User ID Suspend Date

| | |
|---|---|
| FUNCTION: | Define user ID suspend date |
| KEY: | X‘0015’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0006’ |
| SUBFIELD 1 DATA: | CL6‘ ’ |

This field defines the date on which the user ID is to be suspended from further ability to log on to the system. Format is YY.DDD. If omitted, the field defaults to blanks.

## Structured Field Key X‘0016’—Terminal Restrictions

| | |
|---|---|
| FUNCTION: | Define terminal restrictions |
| KEY: | X‘0016’ |
| SUBFIELD COUNT: | X‘0003’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |
| SUBFIELD 2 LENGTH: | X‘0008’ |
| SUBFIELD 2 DATA: | CL8‘ ’ |
| SUBFIELD 3 LENGTH: | X‘0008’ |
| SUBFIELD 3 DATA: | CL8‘ ’ |

Subfields 1, 2 and 3 define the terminals to which this user ID is restricted, if any. Fields can be left blank, or the field omitted if terminal restrictions do not apply to the user ID.

**Note**

The full security exit is responsible for performing the actual terminal restriction processing. Management Services performs no terminal restriction validation when a full security exit is utilized.

## Structured Field Key X‘0017’—Timeout Control

| | |
|---|---|
| FUNCTION: | Define timeout control |
| KEY: | X‘0017’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether the user ID is subject to the Management Services timeout facilities. Valid values are Y or N. If omitted, the default is Y.

## Structured Field Key X‘0018’—Date/Time User ID Last Updated

| | |
|---|---|
| FUNCTION: | Define date/time user ID last updated |
| KEY: | X‘0018’ |
| SUBFIELD COUNT: | X‘0002’ |
| SUBFIELD 1 LENGTH: | X‘0014’ |
| SUBFIELD 1 DATA: | CL20‘ ’ |
| SUBFIELD 2 LENGTH: | X‘0010’ |
| SUBFIELD 2 DATA: | CL16‘ ’ |

The value of subfield 1 represents the date and time that this user ID last logged on to system. The value of subfield 2 represents the date and time that this user ID was last updated.

The format of subfield 1 is a 15 character last logged on date, for example:

```
MON 09-FEB-1987
```

and a 5 character last logged on time, for example:

```
08.30
```

The format of subfield 2 is an 11 character last updated date, for example:

```
09-FEB-1987
```

and a 5 character last updated on time, for example:

```
08.30
```

## Structured Field Key X‘0019’—Multiple Signon Capability

| | |
|---|---|
| FUNCTION: | Define multiple signon capability |
| KEY: | X‘0019’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field determines whether the user ID can log on multiple times from a terminal or MAI environment. Valid values are Y or N only. If omitted, the default is N.

## Structured Field Key X‘001A’ —Group Definition for User

| | |
|---|---|
| FUNCTION: | Define group definition to be used for this user |
| KEY: | X‘001A’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |

This field defines the group definition that is used by this user ID. This definition is read for all functions related to the user ID. The privileges of the group definition are applied to this user ID. Only user information (that is, name, location, phone number and initial OCS command) is specified as unique to this user ID. Valid values are:

- Blank or nulls—indicates that there is no group definition for this user
- Group name—indicates the group definition that is to be used for this user ID

## Structured Field Key X‘001B’—User ID Definition Type

| | |
|---|---|
| FUNCTION: | User ID definition type |
| KEY: | X‘001B’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether this is a Group definition or a User definition. Valid values are Y (group definition) or N (user definition). If omitted, the default is N (user definition).

## Structured Field Key X‘001C’ —User Password Expiry Action Indicator

| | |
|---|---|
| FUNCTION: | User password expiry action indicator |
| KEY: | X‘001C’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether password expiry checking is to be performed for this user ID. Valid values are Y (password renewal at installation determined expiry interval is enforced) or N (no expiry checking is performed). If omitted, the default is Y.

## Structured Field Key X‘0020’ —OCS Access Privilege

| | |
|---|---|
| FUNCTION: | Define OCS access privilege |
| KEY: | X‘0020’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether the user has access to OCS. Valid values are Y or N. If omitted, or an invalid value is specified, the default is N.

## Structured Field Key X‘0021’—Broadcast Services Privilege

| | |
|---|---|
| FUNCTION: | Define Broadcast Services privilege |
| KEY: | X‘0021’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether the user has access to Broadcast Services. Valid values are Y or N. If omitted, or an invalid value is specified, the default is N.

## Structured Field Key X‘0022’ —Network Services Access Privilege

| | |
|---|---|
| FUNCTION: | Define network services access privilege |
| KEY: | X‘0022’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether the user ID has access to the Network Management options. Valid values are Y or N. If omitted, or an invalid value is specified, the default is N. This privilege is a prerequisite for other privileges which are available to the user within Network Services, for example, TCP/IP Services privilege.

This structured field is a prerequisite for the following structured fields:

- SF0026—NEWS Access Privilege
- SF002D—NCS Access Privilege
- SF0090—NCPView Access Privilege
- SF0150—NEWS Reset Privilege
- SF0151—NTS Access Privilege
- SF0530—TCP/IP Services Access Privilege

## Structured Field Key X‘0023’—System Support Privilege

| | |
|---|---|
| FUNCTION: | Define System Support privilege |
| KEY: | X‘0023’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether the user has access to System Support Services. Valid values are Y or N. If omitted, or an invalid value is specified, the default is N.

## Structured Field Key X‘0024’—CAS Panel Maintenance Privilege

| | |
|---|---|
| FUNCTION: | Define CAS Panel Maintenance privilege |
| KEY: | X‘0024’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether the user has access to Panel Maintenance. Valid values are Y or N. If omitted, or an invalid value is specified, the default is N.

## Structured Field Key X‘0025’—FTS Access Privilege

| | |
|---|---|
| FUNCTION: | Define FTS access privilege |
| KEY: | X‘0025’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether the user has access to FTS. Valid values are Y or N. The default is N.

## Structured Field Key X‘0026’—NEWS Access Privilege

| | |
|---|---|
| FUNCTION: | Define NEWS access privilege |
| KEY: | X‘0026’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether the user has access to NEWS. Valid values are Y or N. The default is N.

This field is ignored if the user ID has not been given access privileges to Network Management, that is, if the user ID does not have structured field key X‘0022’.

## Structured Field Key X‘0027’—MAI-FS Access Privilege

| | |
|---|---|
| FUNCTION: | Define MAI-FS access privilege |
| KEY: | X‘0027’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether the user has access to MAI-FS. Valid values are Y or N. If omitted, or an invalid value is specified, the default is N. Specifying Y in this field allows the user to issue SETMODE MAI and use the MAISESS command. In addition, the supplied primary menu offers an option to select the MAI-FS primary menu.

## Structured Field Key X‘0028’—User Services Procedure Name

| | |
|---|---|
| FUNCTION: | Define User Services procedure name |
| KEY: | X‘0028’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |

This field identifies the 8-character name of the NCL procedure to be invoked when the user selects the User Services option from the Primary Menu. The name specified must be a valid NCL procedure name, padded with blanks to 8 characters if required. Imbedded blanks within the name cause truncation.

This field is only valid if the user has access privileges for User Services. If this field is omitted, no User Services procedure name is assigned to the user ID.

## Structured Field Key X‘0029’—User’s NCL Procedure Library

| | |
|---|---|
| FUNCTION: | Define user’s NCL procedure library |
| KEY: | X‘0029’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |

This field provides the DD name in the Management Services JCL (OS/VS) or the Filetype (VM/SP) that defines the NCL procedure library to be used by this user. Whenever the user executes an NCL procedure, this library is searched for the procedure. If omitted, the library defined by the COMMANDS DD statement is used.

## Structured Field Key X‘002A’—UAMS Access Privilege

| | |
|---|---|
| FUNCTION: | Define UAMS access privilege |
| KEY: | X‘002A’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether the user ID has access to the UAMS maintenance functions. Valid values are Y or N. If omitted, or if an invalid value is specified, the default is N.

## Structured Field Key X‘002B’—Operations Management Privilege

| | |
|---|---|
| FUNCTION: | Define Operations Management privilege |
| KEY: | X‘002B’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether the user ID has access to the facilities of Operations Management. Valid values are Y or N. The default is N.

## Structured Field Key X‘002C’—TSO Autologon Privilege

| | |
|---|---|
| FUNCTION: | Define TSO autologon privilege |
| KEY: | X‘002C’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether the user ID can automatically log on to the system using an EIP component, such as a TSO address space, without having to reenter their user ID/password combination. If omitted, the default is N. This field is only valid if you are licensed for EIP.

## Structured Field Key X‘002D’—NCS Access Privilege

| | |
|---|---|
| FUNCTION: | Define NCS access privilege |
| KEY: | X‘002D’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether the user ID has access to the facilities of NCS. Valid values are Y or N. If omitted, or an invalid value is specified, the default is N.

This field is ignored if the user ID has not been given access privileges to Network Management, that is, if the user ID does not have structured field key X‘0022’.

## Structured Field Key X‘002E’ —User’s SPLIT/SWAP Privilege

| | |
|---|---|
| FUNCTION: | Define user’s SPLIT/SWAP privileges |
| KEY: | X‘002E’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether the user ID is entitled to use the Management Services SPLIT/SWAP facility for operating two screen windows. Valid values are Y or N. If omitted, the default is Y.

## Structured Field Key X‘002F’—Library Services Path Name

| | |
|---|---|
| FUNCTION: | Define Library Services path name |
| KEY: | X‘002F’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |

This field defines the user’s library path name. If a library path name is to be used to control the sequence of access to the panels datasets, enter the name of the path name to be used. See the LIBRARY command for details of path name definition. The default value is blanks, indicating that no path name is to be used.

## Structured Field Key X‘0030’—User’s Time Zone Name

| | |
|---|---|
| FUNCTION: | Define user’s time zone name |
| KEY: | X‘0030’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |

This field defines the time zone name for the user. If omitted, the system time zone name is used when the user logs on.

## Structured Field Key X‘0050’—OCS Command Authority Level

| | |
|---|---|
| FUNCTION: | Define OCS command authority level |
| KEY: | X‘0050’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0003’ |
| SUBFIELD 1 DATA: | CL3‘ ’ |

This field defines the command authority level assigned to the user when operating from OCS or when issuing commands within an NCL procedure. Valid values are 0 to 255 inclusive. If omitted, or an invalid value is specified, the default is 0.

## Structured Field Key X‘0051’—OCS Monitor Status

| | |
|---|---|
| FUNCTION: | Define OCS Monitor status |
| KEY: | X‘0051’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether the user is assigned Monitor status when operating in OCS. Valid values are Y or N. If omitted, or an invalid value is specified, the default is N.

## Structured Field Key X‘0052’—NPF Command Member

| | |
|---|---|
| FUNCTION: | Define NPF command member |
| KEY: | X‘0052’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |

This field defines the name of the NPF control member that specifies command partitioning for the user ID. The NPF member name supplied must be padded to 8 characters, if necessary. See Chapter 8, *Partitioning VTAM Networks* for more information about NPF.

## Structured Field Key X‘0053’—MSGPROC Member

| | |
|---|---|
| FUNCTION: | Define MSGPROC member |
| KEY: | X‘0053’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |

This field defines the name of the NCL procedure to be invoked for each OCS window operated by the user. This field only applies if the user requires a MSGPROC NCL procedure to intercept all messages delivered when operating in OCS. The MSGPROC name supplied must be padded to 8 characters, if necessary. No MSGPROC procedure is invoked for the user if this field is omitted.

## Structured Field Key X‘0054’—OCS Mode Initial Command

| | |
|---|---|
| FUNCTION: | Define OCS mode initial command |
| KEY: | X‘0054’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0028’ |
| SUBFIELD 1 DATA: | CL40‘ ’ |

This field defines the command that is executed automatically for this user on entry to OCS. If specified, the field data must be 40 characters long, padded with blanks if required. No validation of the command string is performed. If this field is omitted, no initial command is assigned to the user ID.

## Structured Field Key X‘0055’—PPO Message Receipt Option

| | |
|---|---|
| FUNCTION: | Define PPO message receipt option |
| KEY: | X‘0055’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field determines whether the user is entitled to receive unsolicited VTAM PPO messages, when logged on to OCS. Valid values are Y or N. The default is N.

## Structured Field Key X‘0056’—PPO Severity Level

| | |
|---|---|
| FUNCTION: | Define PPO severity level |
| KEY: | X‘0056’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0004’ |
| SUBFIELD 1 DATA: | CL4‘ ’ |

This field indicates the severity level of PPO messages to be delivered to this user. This field only applies if the user is to have unsolicited VTAM PPO messages delivered when operating in OCS. Valid values are INFO, NORM, WARN, or SER. If omitted, or an invalid value is specified, the default is NORM.

## Structured Field Key X‘0057’—NPF Message Restriction Option

| | |
|---|---|
| FUNCTION: | Define NPF message restriction option |
| KEY: | X‘0057’ |
| SUBFIELD COUNT: | X‘0002’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |
| SUBFIELD 2 LENGTH: | X‘0001’ |
| SUBFIELD 2 DATA: | CL1‘ ’ |

This field defines the NPF message member that specifies the nodes from which the user receives unsolicited VTAM PPO messages. See Chapter 8, *Partitioning VTAM Networks*, for more information about NPF.

This field has 2 subfields. The first subfield specifies the name of the NPF member. The second subfield determines whether the user is to receive undeliverable messages: that is, unsolicited messages that refer to network resources which are outside the user’s normal NPF restrictions but which cannot be delivered to any other user. Valid values are Y or N. The default is N.

This field is ignored if the user ID has not been authorized for PPO message receipt, that is, if the user ID does not have structured field key X‘0055’.

## Structured Field Key X‘0058’—Message Code Value

| | |
|---|---|
| FUNCTION: | Define message code value |
| KEY: | X‘0058’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | XL1‘ ’ |

This field defines the one-byte bit string representing the message level mask for this user if they have a message code set. Any hexadecimal value (00 to FF) is Valid. If omitted, this field defaults to X‘00’.

## Structured Field Key X‘0059’—OCS MSG Message Receipt

| | |
|---|---|
| FUNCTION: | Define OCS MSG message receipt |
| KEY: | X‘0059’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field specifies the default PROFILE MSG command when entering OCS. The default is N.

## Structured Field Key X‘005A’—OCS Unsolicited Message Receipt Option

| | |
|---|---|
| FUNCTION: | Define OCS unsolicited message receipt option |
| KEY: | X‘005A’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field specifies the default PROFILE UNSOL command when entering OCS. The default is N.

## Structured Field Key X‘005B’—Resource List Member

| | |
|---|---|
| FUNCTION: | Define resource list member |
| KEY: | X‘005B’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |

This field defines the resource list member that specifies the combined NPF restrictions for resource display and affect through VTAM commands. If omitted, the field defaults to blanks.

## Structured Field Key X‘005C’—User Timeout (1) Period

| | |
|---|---|
| FUNCTION: | Define user timeout (1) period |
| KEY: | X‘005C’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0006’ |
| SUBFIELD 1 DATA: | CL6‘ ’ |

This field defines the interval used as the first timeout period for an inactive user ID. The format of the interval can be *mmm.ss* (minutes and seconds) or *mmm* (minutes). This field is equivalent to SYSPARMS TOTIME1.

## Structured Field Key X‘005D’—User Timeout (2) Period

| | |
|---|---|
| FUNCTION: | Define user timeout (2) period |
| KEY: | X‘005D’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0006’ |
| SUBFIELD 1 DATA: | CL6‘ ’ |

This field defines the interval used as the second timeout period for an inactive user ID. The format of the interval can be *mmm.ss* (minutes and seconds) or *mmm* (minutes). This field is equivalent to SYSPARMS TOTIME2.

## Structured Field Key X‘005E’—User Timeout (1) Action

| | |
|---|---|
| FUNCTION: | Define user timeout (1) action |
| KEY: | X‘005E’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0028’ |
| SUBFIELD 1 DATA: | CL40‘ ’ |

This field defines the action to be performed when a user ID’s first timeout interval expires. The action can be any command string, for example, LOCK, DISC, CANCEL LU=*. These commands have equivalents in the SYSPARMS timeout actions. A value of NONE can be used to disable the timeout action. This structured field overrides SYSPARMS TOACT1.

## Structured Field Key X‘005F’—User Timeout (2) Action

| | |
|---|---|
| FUNCTION: | Define user timeout (2) action |
| KEY: | X‘005F’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0028’ |
| SUBFIELD 1 DATA: | CL40‘ ’ |

This field defines the action to be performed when a user ID’s second timeout interval expires. The action can be any command string, for example, LOCK, DISC, CANCEL LU=*. These commands have equivalents in the SYSPARMS timeout actions. A value of NONE can be used to disable the timeout action. This structured field overrides SYSPARMS TOACT2.

## Structured Field Key X‘0060’—User’s APPC Access Key

| | |
|---|---|
| FUNCTION: | Define user’s APPC access key |
| KEY: | X‘0060’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0002’ |
| SUBFIELD 1 DATA: | BL2‘00’ |

This field defines the user’s APPC access key. The mask contains 16 bits, representing, from right to left, the access codes 1 to 16. These are used, together with the APPC lock to allow this user to act on behalf of another user, without supplying the target user’s password.

A 16-bit key indicating the APPC access key must be specified. If omitted, the field defaults to zeroes.

## Structured Field Key X‘0061’—User’s APPC Access Lock

| | |
|---|---|
| FUNCTION: | Define user’s APPC access lock |
| KEY: | X‘0061’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0002’ |
| SUBFIELD 1 DATA: | BL2‘00’ |

This field defines the user’s APPC access lock. The key contains 16 bits, representing, form right to left, the access codes 1 to 16. These are used, together with the APPC key, to allow another user to act on behalf of this user, without supplying this user’s password. If omitted, the field defaults to zeroes. Valid value is a 16-bit mask indicating the APPC access lock.

## Structured Field Key X‘0070’ —Installation Attribute Field 1

| | |
|---|---|
| FUNCTION: | Installation Attribute field 1 |
| KEY: | X‘0070’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0020’ |
| SUBFIELD 1 DATA: | CL32‘ ’ |

This field allows specification of data for Installation Attributes assigned to this user ID. The data can be interrogated through NCL. This field corresponds to Installation Attribute field 1 on the Installation Attributes panel in UAMS. If omitted, the field defaults to blanks.

## Structured Field Key X‘0071’—Installation Attribute Field 2

| | |
|---|---|
| FUNCTION: | Installation Attribute field 2 |
| KEY: | X‘0071’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0020’ |
| SUBFIELD 1 DATA: | CL32‘ ’ |

This field allows specification of data for Installation Attributes assigned to this user ID. The data can be interrogated through NCL. This field corresponds to Installation Attribute field 2 on the Installation Attributes panel in UAMS. If omitted, the field defaults to blanks.

## Structured Field Key X‘0072’—Installation Attribute Field 3

| | |
|---|---|
| FUNCTION: | Installation Attribute field 3 |
| KEY: | X‘0072’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0020’ |
| SUBFIELD 1 DATA: | CL32‘ ’ |

This field allows specification of data for Installation Attributes assigned to this user ID. The data can be interrogated through NCL. This field corresponds to Installation Attribute field 3 on the Installation Attributes panel in UAMS. If omitted, the field defaults to blanks.

## Structured Field Key X‘0073’—Installation Attribute Field 4

| | |
|---|---|
| FUNCTION: | Installation Attribute field 4 |
| KEY: | X‘0073’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0020’ |
| SUBFIELD 1 DATA: | CL32‘ ’ |

This field allows specification of data for Installation Attributes assigned to this user ID. The data can be interrogated through NCL. This field corresponds to Installation Attribute field 4 on the Installation Attributes panel in UAMS. If omitted, the field defaults to blanks.

## Structured Field Key X‘0074’—Installation Attribute Field 5

| | |
|---|---|
| FUNCTION: | Installation Attribute field 5 |
| KEY: | X‘0074’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0020’ |
| SUBFIELD 1 DATA: | CL32‘ ’ |

This field allows specification of data for Installation Attributes assigned to this user ID. The data can be interrogated through NCL. This field corresponds to Installation Attribute field 5 on the Installation Attributes panel in UAMS. If omitted, the field defaults to blanks.

## Structured Field Key X‘0075’—Installation Attribute Field 6

| | |
|---|---|
| FUNCTION: | Installation Attribute field 6 |
| KEY: | X‘0075’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0020’ |
| SUBFIELD 1 DATA: | CL32‘ ’ |

This field allows specification of data for Installation Attributes assigned to this user ID. The data can be interrogated through NCL. This field corresponds to Installation Attribute field 6 on the Installation Attributes panel in UAMS. If omitted, the field defaults to blanks.

## Structured Field Key X‘0076’—Installation Attribute Field 7

| | |
|---|---|
| FUNCTION: | Installation Attribute field 7 |
| KEY: | X‘0076’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0020’ |
| SUBFIELD 1 DATA: | CL32‘ ’ |

This field allows specification of data for Installation Attributes assigned to this user ID. The data can be interrogated through NCL. This field corresponds to Installation Attribute field 7 on the Installation Attributes panel in UAMS. If omitted, the field defaults to blanks.

## Structured Field Key X‘0077’—Installation Attribute Field 8

| | |
|---|---|
| FUNCTION: | Installation Attribute field 8 |
| KEY: | X‘0077’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0020’ |
| SUBFIELD 1 DATA: | CL32‘ ’ |

This field allows specification of data for Installation Attributes assigned to this user ID. The data can be interrogated through NCL. This field corresponds to Installation Attribute field 8 on the Installation Attributes panel in UAMS. If omitted, the field defaults to blanks.

## Structured Field Key X‘0078’—Installation Attribute Field 9

| | |
|---|---|
| FUNCTION: | Installation Attribute field 9 |
| KEY: | X‘0078’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0020’ |
| SUBFIELD 1 DATA: | CL32‘ ’ |

This field allows specification of data for Installation Attributes assigned to this user ID. The data can be interrogated through NCL. This field corresponds to Installation Attribute field 9 on the Installation Attributes panel in UAMS. If omitted, the field defaults to blanks.

## Structured Field Key X‘0079’—Installation Attribute Field 10

| | |
|---|---|
| FUNCTION: | Installation Attribute field 10 |
| KEY: | X‘0079’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0020’ |
| SUBFIELD 1 DATA: | CL32‘ ’ |

This field allows specification of data for Installation Attributes assigned to this user ID. The data can be interrogated through NCL. This field corresponds to Installation Attribute field 10 on the Installation Attributes panel in UAMS. If omitted, the field defaults to blanks.

## Structured Field Key X‘0080’—User’s Access to INFO/MASTER Maintenance Functions

| | |
|---|---|
| FUNCTION: | Define user’s access to INFO/MASTER maintenance functions |
| KEY: | X‘0080’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether the user ID has access to INFO/MASTER maintenance functions. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0081’—User’s Access to Information Services

| | |
|---|---|
| FUNCTION: | Define user’s access to Information Services |
| KEY: | X‘0081’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether the user ID has access to Information Services. Valid values are Y or N. If omitted, the default is Y.

## Structured Field Key X‘0090’—User’s Access to NCPView

| | |
|---|---|
| FUNCTION: | Define user’s access to NCPView |
| KEY: | X‘0090’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to NCPView. Valid values are 0 (not authorized to access NCPView), and 1 (authorized to browse).

This field is ignored if the user ID has not been given access privileges to Network Management, that is, if the user ID does not have structured field key X‘0022’.

## Structured Field X‘0150’—Define NEWS Reset Privilege

| | |
|---|---|
| FUNCTION: | Define NEWS reset privilege |
| KEY: | X‘0150’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

A user who has NEWS access can be authorized to reset (delete) database records. Valid values are Y or N. The default is N.

Specifying Y in this field enables the user to reset statistics within the Summary Statistics Subsystem, and results in the &NEWSRSET system variable being set to YES when tested within an NCL procedure invoked by the user.

This field is ignored if the user ID has not been given access privileges to Network Management and to NEWS, that is, if the user ID does not have structured field keys X‘0022’ and X‘0026’.

## Structured Field X‘0151’—Define NTS Access Privilege

| | |
|---|---|
| FUNCTION: | Define NTS access privilege |
| KEY: | X‘0151’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user can access the facilities of the Network Tracking System. Valid values are Y or N. The default is N.

This field is ignored if NTS is not configured on your system.

This field is ignored if the user ID has not been given access privileges to Network Management, that is, if the user ID does not have structured field key X‘0022’.

## Structured Field Key X‘0180’—User’s AOM Message Delivery and Routing Codes

| | |
|---|---|
| FUNCTION: | Define user’s AOM message delivery and routing codes |
| KEY: | X‘0180’ |
| SUBFIELD COUNT: | X‘0002’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |
| SUBFIELD 2 LENGTH: | X‘0010’ or X’0002’ |
| SUBFIELD 2 DATA: | BL16‘ ’ |

This field defines whether a user ID is entitled to receive AOM messages.

Subfield 1 indicates if the user is entitled to receive AOM messages while in OCS mode. The value must be Y or N. Invalid values are treated as N.

Subfield 2 is a 128-bit mask that indicates the routing codes for which the user is entitled to receive messages. The mask contains 128 bits, which are used to represent routing codes 1 to 128 inclusive, from left to right. All 16 bytes are supported, providing routing codes 1 to 128.

For example, code the value of subfield 2 as X’8200000000000000’ to indicate the user is authorized for routing codes 1 and 7—the binary equivalent of X’8200000000000000’ is B’10000010...0000’, which has the 1-bit on for routing codes 1 and 7 (reading from left to right).

## Structured Field Key X‘0181’—User’s AOM MVS SYSCMD Console Authority

| | |
|---|---|
| FUNCTION: | Define user’s AOM MVS SYSCMD console authority |
| KEY: | X‘0181’ |
| SUBFIELD COUNT: | X‘0004’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |
| SUBFIELD 2 LENGTH: | X‘0001’ |
| SUBFIELD 2 DATA: | CL1‘ ’ |
| SUBFIELD 3 LENGTH: | X‘0001’ |
| SUBFIELD 3 DATA: | CL1‘ ’ |
| SUBFIELD 4 LENGTH: | X‘0001’ |
| SUBFIELD 4 DATA: | CL1‘ ’ |

This field defines a user ID’s console and command authority for AOM MVS SYSCMD.

The following is a description for each of the subfields of this structured field:

- SUBFIELD 1—AOM MVS SYSCMD Console Authority level. The only values for the subfield are:

  I Information only console
  M Master console
  P Pseudo-master console
  C Subfields 2 to 4 indicate the console command authority for this user ID

  Any other values in this subfield are ignored.

- SUBFIELD 2—authority to issue SYSTEM commands
- SUBFIELD 3—authority to issue IO commands
- SUBFIELD 4—authority to issue CONSOLE commands

For subfields 2 to 4 you can only enter Y or N. Management Services assumes N if a field is omitted or invalid.

## Structured Field Key X‘0182’ —User’s AOM MSG Level

| | |
|---|---|
| FUNCTION: | Define user’s AOM MSG level |
| KEY: | X‘0182’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | B‘*xxxxxx*00’ |

This field defines the 8-bit mask describing the user ID’s MSG level. Valid bit settings are as follows:

| | |
|---|---|
| 1... .... | Receive WTORs |
| .1.. .... | Receive Immediate Action |
| ..1. .... | Receive Critical Eventual Action |
| ...1 .... | Receive Eventual Action |
| .... 1... | Receive Informational |
| .... .1.. | Receive Console Broadcasts |
| .... ..11 | Reserved |

## Structured Field Key X‘0183’—User’s AOM VM SYSCMD Authority

| | |
|---|---|
| FUNCTION: | Define user’s AOM VM SYSCMD authority |
| KEY: | X‘0183’ |
| SUBFIELD COUNT: | X‘0002’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |
| SUBFIELD 2 LENGTH: | X‘0001’ |
| SUBFIELD 2 DATA: | CL1‘ ’ |

This field defines a user ID’s AOM VM SYSCMD authority.

The first subfield controls whether the user is authorized to issue SYSCMD DEST=PROP. Valid values are Y or N. Invalid values are treated as N.

The second subfield controls whether the user is authorized to issue SYSCMD DEST=GCS. Valid values are Y or N. Invalid values are treated as N.

This structured field is ignored by AOM/MVS. It allows shared security profile for MVS and VM systems.

## Structured Field Key X‘0185’ —User’s AOM VOS3/JSS4 SYSCMD Command Authority

| | |
|---|---|
| FUNCTION: | Define user’s AOM VOS3/JSS4 SYSCMD command authority |
| KEY: | X‘0185’ |
| SUBFIELD COUNT: | X‘000’ |
| SUBFIELD 1 LENGTH: | X‘0001’ or X‘0002’ |
| SUBFIELD 1 DATA: | CL1‘*n* ’ or CL2‘*nn*’ |

This field provides the VOS3/JSS4 command authority level for SYSCMD. It is ignored by AOM/VM and AOM/MVS when not in a VOS3/JSS4 environment. This allows shared security profile across all systems.

**Note**

The UAMS panel that displays AOM/MVS SYSCMD authority always shows this field and allows update. The field is not used by SYSCMD unless the system is actually VOS3/JSS4.

## Structured Field Key X‘0200’—User’s MAI-FS Privilege Class

| | |
|---|---|
| FUNCTION: | Define user’s MAI-FS privilege class |
| KEY: | X‘0200’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines the MAI-FS privilege class for the user ID. Valid values are A, B, C, or D. If this field is omitted, the default is D (at user logon).

## Structured Field Key X‘0201’—User’s MAI-FS Model User ID

| | |
|---|---|
| FUNCTION: | Define user’s MAI-FS model user ID |
| KEY: | X‘0201’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |

This field supplies an MAI-FS user ID on whose stored session definitions this user ID is to be modeled. If this field is omitted, the field is left blank.

## Structured Field Key X‘0202’—User’s MAI-FS A and E Command Capability

| | |
|---|---|
| FUNCTION: | Define user’s MAI-FS A and E command capability |
| KEY: | X‘0202’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field specifies whether a user is entitled to use the A and E commands. Valid values are Y and N. If this field is omitted, the default is N.

## Structured Field Key X‘0500’—User’s PSM Primary Menu Access

| | |
|---|---|
| FUNCTION: | Define user’s PSM primary menu access |
| KEY: | X‘0500’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether the user is authorized to access the PSM : Primary Menu. Valid values are 0 (not authorized) and 1 (authorized). If this field is omitted, the default is 1 (authorized).

## Structured Field Key X‘0501’—User’s PSM Maintenance Access

| | |
|---|---|
| FUNCTION: | Define user’s PSM maintenance access |
| KEY: | X‘0501’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines the level of access a user has to PSM printer, form, and setup definitions, and to default printer assignment. Valid values are 0 (not authorized), 1 (browse access only), 2 (browse, add, and update access), and 3 (browse, add, update, and delete access). If this field is omitted, the default is 1 (browse).

## Structured Field Key X‘0502’—User’s PSM Administration Access

| | |
|---|---|
| FUNCTION: | Define user’s PSM administration access |
| KEY: | X‘0502’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines the level of access a user has to PSM administration functions. Valid values are 0 (not authorized), 1 (browse, and update default definitions), 2 (browse and update default definitions, and clear the spool). If this field is omitted, the default is 0 (not authorized).

## Structured Field Key X‘0503’—User’s PSM Ability to Change Print Request Priority

| | |
|---|---|
| FUNCTION: | Define user’s PSM ability to change print request priority |
| KEY: | X‘0503’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines a user’s ability to change the priority for a PSM print request. Valid values are 0 (not authorized), and 1 (authorized). If this field is omitted, the default is 0 (not authorized).

## Structured Field Key X‘0504’—User’s PSM Queue Access for All Print Output

| | |
|---|---|
| FUNCTION: | Define user’s PSM queue access for all print output |
| KEY: | X‘0504’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines a user’s queue access to other user’s PSM print output. Valid values are 0 (not authorized to view the output queue for other users), 1 (authorized to view the output queue for other users), 2 (authorized to browse other user output), 3 (authorized to browse, modify, and release other user output), 4 (authorized to browse, modify, release, hold, and delete other user output). If this field is omitted, the default is 1.

## Structured Field Key X‘0505’—User’s PSM Queue Access for Their Own Print Output

| | |
|---|---|
| FUNCTION: | Define user’s PSM queue access for their own print output |
| KEY: | X‘0505’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines the user’s queue access to their own PSM print output. Valid values are 0 (not authorized to view their output queue), 1 (authorized to view their output queue), 2 (authorized to browse their output), 3 (authorized to browse, modify, and release their output), 4 (authorized to browse, modify, release, hold, and delete their output). If this field is omitted, the default is 4.

## Structured Field Key X‘0510’ —User’s Panel Command Access Authority

| | |
|---|---|
| FUNCTION: | Define user’s panel command access authority |
| KEY: | X‘0510’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user is entitled to enter EXEC or CMD in the COMMAND ===> and Select Option ===> input fields on panels. Valid values are Y and N. The default is N. If the user has OCS authority, this field is automatically set to Y and cannot be altered.

## Structured Field Key X‘0511’ —User’s Management Services Access

| | |
|---|---|
| FUNCTION: | Define user’s Management Services access |
| KEY: | X‘0511’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field indicates whether the user has access to Management Services functions (that is, UAMS, Broadcast Services, OS Support, System Support Services). Valid values are Y and N. The default is N.

## Structured Field Key X‘0520’—User’s Notification Details (First Rule)

| | |
|---|---|
| FUNCTION: | Define user’s notification details (first rule) |
| KEY: | X‘0520’ |
| SUBFIELD COUNT: | X‘0008’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |
| SUBFIELD 2 LENGTH: | X‘0004’ |
| SUBFIELD 2 DATA: | CL4‘ ’ |
| SUBFIELD 3 LENGTH: | X‘0007’ |
| SUBFIELD 3 DATA: | CL7‘ ’ |
| SUBFIELD 4 LENGTH: | X‘0004’ |
| SUBFIELD 4 DATA: | CL4‘ ’ |
| SUBFIELD 5 LENGTH: | X‘0004’ |
| SUBFIELD 5 DATA: | CL4‘ ’ |
| SUBFIELD 6 LENGTH: | X‘0020’ |
| SUBFIELD 6 DATA: | CL32‘ ’ |
| SUBFIELD 7 LENGTH: | X‘0008’ |
| SUBFIELD 7 DATA: | CL8‘ ’ |
| SUBFIELD 8 LENGTH: | X‘0040’ |
| SUBFIELD 8 DATA: | CL64‘ ’ |

This field defines the mode or method by which the user wishes to be notified of an event that the user has been nominated to receive. These notification details are used by various applications. Broadcast services also supports the use of the UAMS Notification Details when sending a broadcast to a specific user via the N option.

This structured field has multiple subfields containing the following information:

SUBFIELD 1—Notification Mode
SUBFIELD 2—Notification Domain
SUBFIELD 3—Day of Notification
SUBFIELD 4—Low Time
SUBFIELD 5—High Time
SUBFIELD 6—User ID
SUBFIELD 7—Exit Name
SUBFIELD 8—Parameters

If omitted, the fields are left blank.

## Structured Field Key X‘0521’—User’s Notification Details (Second Rule)

| | |
|---|---|
| FUNCTION: | Define user’s notification details (second rule) |
| KEY: | X‘0521’ |
| SUBFIELD COUNT: | X‘0008’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |
| SUBFIELD 2 LENGTH: | X‘0004’ |
| SUBFIELD 2 DATA: | CL4‘ ’ |
| SUBFIELD 3 LENGTH: | X‘0007’ |
| SUBFIELD 3 DATA: | CL7‘ ’ |
| SUBFIELD 4 LENGTH: | X‘0004’ |
| SUBFIELD 4 DATA: | CL4‘ ’ |
| SUBFIELD 5 LENGTH: | X‘0004’ |
| SUBFIELD 5 DATA: | CL4‘ ’ |
| SUBFIELD 6 LENGTH: | X‘0020’ |
| SUBFIELD 6 DATA: | CL32‘ ’ |
| SUBFIELD 7 LENGTH: | X‘0008’ |
| SUBFIELD 7 DATA: | CL8‘ ’ |
| SUBFIELD 8 LENGTH: | X‘0040’ |
| SUBFIELD 8 DATA: | CL64‘ ’ |

This field defines the mode or method by which the user wishes to be notified of an event that the user has been nominated to receive. These notification details are used by various applications. Broadcast services also supports the use of the UAMS Notification Details when sending a broadcast to a specific user via the N option.

This structured field has multiple subfields containing the following information:

SUBFIELD 1—Notification Mode
SUBFIELD 2—Notification Domain
SUBFIELD 3—Day of Notification
SUBFIELD 4—Low Time
SUBFIELD 5—High Time
SUBFIELD 6—User ID
SUBFIELD 7—Exit Name
SUBFIELD 8—Parameters

If omitted, the fields are left blank.

## Structured Field Key X‘0522’—User’s Notification Details (Third Rule)

| | |
|---|---|
| FUNCTION: | Define user’s notification details (third rule) |
| KEY: | X‘0522’ |
| SUBFIELD COUNT: | X‘0008’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |
| SUBFIELD 2 LENGTH: | X‘0004’ |
| SUBFIELD 2 DATA: | CL4‘ ’ |
| SUBFIELD 3 LENGTH: | X‘0007’ |
| SUBFIELD 3 DATA: | CL7‘ ’ |
| SUBFIELD 4 LENGTH: | X‘0004’ |
| SUBFIELD 4 DATA: | CL4‘ ’ |
| SUBFIELD 5 LENGTH: | X‘0004’ |
| SUBFIELD 5 DATA: | CL4‘ ’ |
| SUBFIELD 6 LENGTH: | X‘0020’ |
| SUBFIELD 6 DATA: | CL32‘ ’ |
| SUBFIELD 7 LENGTH: | X‘0008’ |
| SUBFIELD 7 DATA: | CL8‘ ’ |
| SUBFIELD 8 LENGTH: | X‘0040’ |
| SUBFIELD 8 DATA: | CL64‘ ’ |

This field defines the mode or method by which the user wishes to be notified of an event that the user has been nominated to receive. These notification details are used by various applications. Broadcast services also supports the use of the UAMS Notification Details when sending a broadcast to a specific user via the N option.

This structured field has multiple subfields containing the following information:

SUBFIELD 1—Notification Mode
SUBFIELD 2—Notification Domain
SUBFIELD 3—Day of Notification
SUBFIELD 4—Low Time
SUBFIELD 5—High Time
SUBFIELD 6—User ID
SUBFIELD 7—Exit Name
SUBFIELD 8—Parameters

If omitted, the fields are left blank.

## Structured Field Key X‘0523’—User’s Notification Details (Fourth Rule)

| | |
|---|---|
| FUNCTION: | Define user’s notification details (fourth rule) |
| KEY: | X‘0523’ |
| SUBFIELD COUNT: | X‘0008’ |
| SUBFIELD 1 LENGTH: | X‘0008’ |
| SUBFIELD 1 DATA: | CL8‘ ’ |
| SUBFIELD 2 LENGTH: | X‘0004’ |
| SUBFIELD 2 DATA: | CL4‘ ’ |
| SUBFIELD 3 LENGTH: | X‘0007’ |
| SUBFIELD 3 DATA: | CL7‘ ’ |
| SUBFIELD 4 LENGTH: | X‘0004’ |
| SUBFIELD 4 DATA: | CL4‘ ’ |
| SUBFIELD 5 LENGTH: | X‘0004’ |
| SUBFIELD 5 DATA: | CL4‘ ’ |
| SUBFIELD 6 LENGTH: | X‘0020’ |
| SUBFIELD 6 DATA: | CL32‘ ’ |
| SUBFIELD 7 LENGTH: | X‘0008’ |
| SUBFIELD 7 DATA: | CL8‘ ’ |
| SUBFIELD 8 LENGTH: | X‘0040’ |
| SUBFIELD 8 DATA: | CL64‘ ’ |

This field defines the mode or method by which the user wishes to be notified of an event that the user has been nominated to receive. These notification details are used by various applications. Broadcast services also supports the use of the UAMS Notification Details when sending a broadcast to a specific user via the N option.

This structured field has multiple subfields containing the following information:

SUBFIELD 1—Notification Mode
SUBFIELD 2—Notification Domain
SUBFIELD 3—Day of Notification
SUBFIELD 4—Low Time
SUBFIELD 5—High Time
SUBFIELD 6—User ID
SUBFIELD 7—Exit Name
SUBFIELD 8—Parameters

If omitted, the fields are left blank.

## Structured Field Key X‘0530’—TCP/IP Services Access Privilege

| | |
|---|---|
| FUNCTION: | Defines a user’s TCP/IP Services authority level. |
| KEY: | X‘0530’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

Defines the authority level for a user using TCP/IP Services. Valid values are 0 (not authorized to access TCP/IP Services), 1 (browse only authority), and 2 (full update authority which is necessary to: run Obeyfiles, drop connections, perform packet tracing, start and stop devices, and use the SNMP Set option). If omitted, the default is 0.

This field is ignored if the user ID has not been given access privileges to Network Management, that is, if the user ID does not have structured field key X‘0022’.

## Structured Field Key X‘0550’—User’s Report Writer Primary Menu Access

| | |
|---|---|
| FUNCTION: | Define user’s Report Writer primary menu access |
| KEY: | X‘0550’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID is authorized to access the Report Writer primary menu. Valid values are 0 (not authorized), and 1 (authorized). If omitted, the default is 1.

## Structured Field Key X‘0551’—User’s Report Writer Administration Access

| | |
|---|---|
| FUNCTION: | Define user’s Report Writer administration access |
| KEY: | X‘0551’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID is authorized to access Report Writer table maintenance administration functions. Valid values are 0 (not authorized), and 1 (authorized). If omitted, the default is 0.

## Structured Field Key X‘0552’—User’s Report Writer Maintenance Access

| | |
|---|---|
| FUNCTION: | Define user’s Report Writer maintenance access |
| KEY: | X‘0552’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user is authorized to access the Report Writer report definition menu. Valid values are 0 (not authorized for menu access), and 1 (menu access authorized). If omitted, the default is 1.

## Structured Field Key X‘0553’—User’s Report Writer Public Report Access

| | |
|---|---|
| FUNCTION: | Define user’s Report Writer public report access |
| KEY: | X‘0553’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines the level of access a user ID has to public reports. Valid values are 0 (not authorized), 1 (authorized to generate, browse, and copy), 2 (authorized to generate, browse, copy, add, and update), 3 (authorized to generate, browse, copy, add, update, and delete). If omitted, the default is 1.

## Structured Field Key X‘0554’—User’s Report Writer Access to Their Own Reports

| | |
|---|---|
| FUNCTION: | Define user’s Report Writer access to their own reports |
| KEY: | X‘0554’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines the level of access a user ID has to their own reports. Valid values are 0 (not authorized), 1 (authorized to generate, browse, and copy), 2 (authorized to generate, browse, copy, add, and update), 3 (authorized to generate, browse, copy, add, update, and delete). If omitted, the default is 3.

## Structured Field Key X‘0555’—User’s Report Writer Private Report Access for All Users

| | |
|---|---|
| FUNCTION: | Define user’s Report Writer private report access for all users |
| KEY: | X‘0555’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines the level of access a user ID has to private reports for all users. Valid values are 0 (not authorized=), 1 (authorized to generate, browse, and copy), 2 (authorized to generate, browse, copy, add, and update), 3 (authorized to generate, browse, copy, add, update, and delete). If omitted, the default is 0.

## Structured Field Key X‘0556’—User’s Report Writer Schedule Maintenance Access

| | |
|---|---|
| FUNCTION: | Define user’s Report Writer schedule maintenance access |
| KEY: | X‘0556’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines the level of access a user ID has to Report Writer schedule definitions. Valid values are 0 (not authorized), 1 (authorized to browse), 2 (authorized to browse, add, and update), 3 (authorized to generate, browse, add, update, and delete), and 4 (authorized for the Stop and Start functions of schedule processing). If omitted, the default is 1.

## Structured Field Key X‘0580’—User’s Access to SOLVE:NetMail

| | |
|---|---|
| FUNCTION: | Define user’s access to SOLVE:NetMail |
| KEY: | X‘0580’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to SOLVE:NetMail. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0600’—User’s Access to SOLVE:Central Applications

| | |
|---|---|
| FUNCTION: | Define user’s access to the SOLVE:Central applications |
| KEY: | X‘0600’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to SOLVE:Central applications. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0601’—User’s Access to Managed Objects Development Services (MODS)

| | |
|---|---|
| FUNCTION: | Define user’s access to Managed Objects Development Services (MODS) |
| KEY: | X‘0601’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to MODS. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0602’—User’s Access to the MODS Application Register

| | |
|---|---|
| FUNCTION: | Define user’s access to the MODS application register |
| KEY: | X‘0602’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to the MODS Application Register. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0603’—User’s Access to the MODS Administration Function

| | |
|---|---|
| FUNCTION: | Define user’s access to the MODS administration function |
| KEY: | X‘0603’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to the MODS Administration function. Valid values are Y or N. The default is N.

## Structured Field Key X‘0604’—User’s SOLVE:Central Primary Menu

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Central primary menu |
| KEY: | X‘0604’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0006’ |
| SUBFIELD 1 DATA: | CL6‘ ’ |

This field defines the SOLVE:Central primary menu to be displayed for a user ID. The value must be a valid menu identifier. Menu identifiers are made up of a 3 character application ID followed by 3 numeric characters in the range 001 to 225.

## Structured Field Key X‘0605’—User’s Object Services Access

| | |
|---|---|
| FUNCTION: | Define user’s Object Services access |
| KEY: | X‘0605’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to Object Services support functions. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0606’—User’s Object Services Object Class Specification Access

| | |
|---|---|
| FUNCTION: | Define user’s Object Services object class specification access |
| KEY: | X‘0606’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to Object Services Object Class Specification function. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0607’—User’s Access to Mapping Services

| | |
|---|---|
| FUNCTION: | Define user’s access to Mapping Services |
| KEY: | X‘0607’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to Mapping Services. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0608’—User’s SOLVE:Central Security Profile

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Central security profile |
| KEY: | X‘0608’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘000C’ |
| SUBFIELD 1 DATA: | CL12‘ ’ |

Defines the Object Services Security Profile for the Enterprise classes.

## Structured Field Key X‘0609’—User’s Object Services Security Access

| | |
|---|---|
| FUNCTION: | Define user’s Object Services security access |
| KEY: | X‘0609’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to Object Services. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0610’—User’s CAS Menu Maintenance Access

| | |
|---|---|
| FUNCTION: | Defines a user’s CAS menu maintenance access |
| KEY: | X‘0610’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to CAS Menu Maintenance. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0611’—User’s CAS List Maintenance Access

| | |
|---|---|
| FUNCTION: | Define user’s CAS list maintenance access |
| KEY: | X‘0611’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to CAS List Maintenance. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0612’—User’s CAS Panel Domain Maintenance Access

| | |
|---|---|
| FUNCTION: | Define user’s CAS panel domain maintenance access |
| KEY: | X‘0612’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to CAS Panel Domain Maintenance. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0613’—User’s CAS Help Maintenance Access

| | |
|---|---|
| FUNCTION: | Define user’s CAS help maintenance access |
| KEY: | X‘0613’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to CAS Help Maintenance. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0614’—User’s CAS Message Maintenance Access

| | |
|---|---|
| FUNCTION: | Define user’s CAS message maintenance access |
| KEY: | X‘0614’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to CAS Message Maintenance. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0615’—User’s CAS Table Maintenance Access

| | |
|---|---|
| FUNCTION: | Define user’s CAS table maintenance access |
| KEY: | X‘0615’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to CAS Table Maintenance. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0616’—User’s CAS Criteria Maintenance Access

| | |
|---|---|
| FUNCTION: | Define user’s CAS criteria maintenance access |
| KEY: | X‘0616’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to CAS Criteria Maintenance. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0617’—User’s CAS Command Maintenance Access

| | |
|---|---|
| FUNCTION: | Define user’s CAS command maintenance access |
| KEY: | X‘0617’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0001’ |
| SUBFIELD 1 DATA: | CL1‘ ’ |

This field defines whether a user ID has access to CAS Command Maintenance. Valid values are Y or N. If omitted, the default is N.

## Structured Field Key X‘0620’—User’s SOLVE:Problem Security Profile

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Problem security profile |
| KEY: | X‘0620’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘000C’ |
| SUBFIELD 1 DATA: | CL12‘ ’ |

Defines the Object Services Security Profile to be used for the SOLVE:Problem user.

## Structured Field Key X‘0621’—User’s SOLVE:Problem Primary Menu ID

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Problem primary menu ID |
| KEY: | X‘0621’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0006’ |
| SUBFIELD 1 DATA: | CL6‘ ’ |

This field defines the SOLVE:Problem primary menu to be displayed for a user ID. The value must be a valid menu identifier. Menu identifiers are made up of a 3-character application ID followed by 3 numeric characters in the range 001 to 225.

## Structured Field Key X‘0640’—User’s SOLVE:Change Security Profile

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Change security profile |
| KEY: | X‘0640’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘000C’ |
| SUBFIELD 1 DATA: | CL12‘ ’ |

Defines the Object Services Security Profile for the SOLVE:Change user.

## Structured Field Key X‘0641’—User’s SOLVE:Change Primary Menu ID

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Change primary menu ID |
| KEY: | X‘0641’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0006’ |
| SUBFIELD 1 DATA: | CL6‘ ’ |

Defines the SOLVE:Change primary menu to be displayed for a user ID. The value must be a valid menu identifier. Menu Identifiers are made up of a 3-character application ID followed by 3 numeric characters in the range 001 to 225.

## Structured Field Key X‘0660’—User’s SOLVE:Asset Security Profile

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Asset security profile |
| KEY: | X‘0660’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘000C’ |
| SUBFIELD 1 DATA: | CL12‘ ’ |

Defines the Object Services Security Profile for the SOLVE:Asset user.

## Structured Field Key X‘0661’ —User’s SOLVE:Asset Primary Menu ID

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Asset primary menu ID |
| KEY: | X‘0661’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0006’ |
| SUBFIELD 1 DATA: | CL6‘ ’ |

Defines the SOLVE:Asset primary menu to be displayed for a user ID. The value must be a valid menu identifier. Menu identifiers are made up of a 3-character application ID followed by 3 numeric characters in the range 001 to 225.

## Structured Field Key X‘0680’—User’s SOLVE:Configuration Security Profile

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Configuration security profile |
| KEY: | X‘0680’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘000C’ |
| SUBFIELD 1 DATA: | CL12‘ ’ |

Defines the Object Services Security Profile for the SOLVE:Configuration user.

## Structured Field Key X‘0681’—User’s SOLVE:Configuration Primary Menu ID

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Configuration primary menu ID |
| KEY: | X‘0681’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0006’ |
| SUBFIELD 1 DATA: | CL6‘ ’ |

Defines the SOLVE:Configuration primary menu to be displayed for a user ID. The value must be a valid menu identifier. Menu identifiers are made up of a 3 character application ID followed by 3 numeric characters in the range 001 to 225.

## Structured Field Key X‘0700’—User’s SOLVE:Problem Helpdesk Security Profile

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Problem Helpdesk security profile |
| KEY: | X‘0700’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘000C’ |
| SUBFIELD 1 DATA: | CL12‘ ’ |

Defines the Object Services Security Profile for the SOLVE:Problem Helpdesk user.

## Structured Field Key X‘0701’—User’s SOLVE:Problem Helpdesk Primary Menu ID

| | |
|---|---|
| FUNCTION: | Define user’s SOLVE:Problem Helpdesk primary menu ID |
| KEY: | X‘0701’ |
| SUBFIELD COUNT: | X‘0001’ |
| SUBFIELD 1 LENGTH: | X‘0006’ |
| SUBFIELD 1 DATA: | CL6‘ ’ |

Defines the SOLVE:Problem Helpdesk primary menu to be displayed for a user ID. The value must be a valid menu identifier. Menu identifiers are made up of a 3-character application ID followed by 3 numeric characters in the range 001 to 225.

# H

# Command Authority Levels

This appendix provides information about command authority levels. It lists all the Management Services (MS) commands and the default authority level required to execute each one.

## About Command Authority Levels

Each Management Services command has an assigned authority level in the range 0 to 255. For a user to be authorized to execute a particular command, their corresponding command authority level must be equal to or higher than the authority level assigned to the command. Command authority levels for users are allocated in their user ID definition.

Default command authority levels have been allocated to each Management Services command according to the importance or power associated with each command. For example, VTAM display commands have an authority level of 0. A user with a corresponding command authority level of 0 can execute display commands, however, they cannot execute commands that might affect or change the status of the network, as those commands require a higher authority.

The default command authority level of Management Services commands can be changed. For more information about this see Chapter 2, *Customizing Management Services*.

## Command Authority Summary Table

The following table describes the available commands and their default authority level. Where a special authority applies to certain operands of a command, the authority levels are expressed as normal/special, for example, 0/2. These are the commands for which the *opauth* sub-parameter of the SYSPARMS CMDAUTH operand can be used. See Appendix D, *SYSPARMS Operands* for more information about using this operand.

Commands with an authority level of 'T' are controlled by the TCP/IP access level.

For detailed information about each Management Services command, see the *Management Services Command Reference.*

| Command | Description | Auth |
|---|---|---|
| ACT | Activates a VTAM network node | 1 |
| ACTLOGON | Activates previously suspended logon paths | 3 |
| ALLOCATE | Allocates a cataloged or sysout or VSAM dataset to Management Services | 0/3 |
| AOM GO | Restarts the local AOM operating system after a *pause* | 2 |
| AOM NEWTAB | Compiles and (optionally) loads a new AOM Screening Table | 2 |
| AOM PAUSE | Suspends the local AOM operating system interface | 2 |
| AOM START | Starts the local AOM operating system interface | 2 |
| AOM STOP | Terminates the local AOM operating system interface | 2 |
| AT | Issues a command(s) or a message AT the specified time of day | 0/2 |
| ATF DEFINE | Defines a new translation name. | 2 |
| ATF LOAD | Loads translation tables for a specific translation name into storage. | 2 |
| ATF UNLOAD | Unloads translation tables for a specific translation name. | 2 |
| AUTOHOLD | Determines the OCS screen display line mode | 0 |
| CANCEL | Cancels a nominated user or user region or window | 0/1 |
| CLEAR | Clears the current logical screen window | 0 |

| Command | Description | Auth |
|---|---|---|
| CLSDST | Closes the session(s) with the specified terminal(s) | 1/0 |
| CNM | Starts and stops the VTAM CNM interface | 3 |
| CNMTRACE | Defines CNM records to be traced | 3 |
| CONNECT | Connects a terminal to Management Services using XNF | 1 |
| CS- | Retrieves the command from the bottom of the command stack | 0 |
| CS+ | Retrieves the command from the top of the command stack | 0 |
| D | Displays a specific VTAM resource | 0 |
| DEBUG | Controls the execution and debugging of NCL processes | 0/3 |
| DEFALIAS | Defines an alias entry for the Alias Name Translation Facility of NEWS | 4 |
| DEFCLASS | Defines RTM, SAW, Session or Resource classes | 4 |
| DEFLINK | Maintains the Dynamic Link Table, or DLT, used by LU6.2 services | 3 |
| DEFLOGON | Defines the logon information for the EASINET and MAI features | 4 |
| DEFMODE | Defines an entry in the Mode Control Table used by LU6.2 services | 4 |
| DEFMSG | Defines delivery options for PPO messages | 4 |
| DEFOPSET | Defines an entry in the OSCT table | 4 |
| DEFTERM | Defines terminal type (VOS3 only) | 2 |
| DEFTRANS | Defines an entry in the or TCT used by LU6.2 services | 4 |
| DELALIAS | Deletes an alias entry used by the Alias Name Translation Facility | 4 |
| DELCLASS | Deletes an NTS class definition | 4 |
| DELLINK | Deletes a dynamic link definition | 3 |
| DELLOGON | Deletes an entry form EASINET/MAI appl table | 4 |
| DELMODE | Deletes an APPC MCT table entry | 4 |
| DELOPSET | Deletes an APPC Option Set Control Table (OSCT) entry | 4 |

| Command | Description | Auth |
|---|---|---|
| DELTERM | Deletes terminal definition (VOS3 only) | 2 |
| DELTRANS | Deletes an APPC Transaction Control Table (TCT) entry | 4 |
| DISCONN | Disconnects either the current or the nominated processing region | 0/1 |
| DNR | Defines or modifies parameters for the Domain Name Resolver (DNR) | 4 |
| DOMAIN | Uniquely defines a domain. | 3 |
| EDB | Defines, starts, stops, or deletes an EDB connection | 4 |
| EDIT | Invokes Panel maintenance from an OCS window or NCL procedure | 0 |
| END | Terminates a paused NCL process without further processing | 0/2 |
| ENTER | Defines the ENTER key | 0 |
| EQUATE | Assigns a text value to a unique string | 0 |
| EVERY | Issues a command or series of commands at a specified time frequency | 0/2 |
| EXEC | Schedules an NCL process for serial execution | 0 |
| F | Issues a VTAM modify (F) command | 1 |
| FLUSH | Terminates an NCL process without further processing | 0/2 |
| FORCE | Forcibly inactivates a VTAM network node | 1 |
| FSPROC | Executes an NCL procedure in OCS fullscreen mode | 0 |
| FSTOP | Forces an immediate shutdown of the system | 3 |
| FTS | Involves File Transmission Services | 0 |
| FTSINIT | Modifies the status of FTS initiators | 0 |
| FTSMOD | Modifies the status of incoming or outgoing FTS transmissions | 0 |
| GO | Resumes processing for a paused NCL process | 0/2 |
| INACT | Inactivates a VTAM network node | 1 |
| INTQUE | Passes data to an NCL procedure’s &INTREAD statement | 0/2 |

| Command | Description | Auth |
|---|---|---|
| ISR | Changes the status of ISR (Inter System Routing) conversations | 2 |
| JOURNAL | Controls the Management Services journal dataset | 2 |
| K | Clears the current OCS window | 0 |
| LIBPATH | Controls library path definitions | 4 |
| LIBRARY | Controls library definitions | 4 |
| LINK | Defines or changes the status of the INMC/NVS/APPC links | 3 |
| LINK START | Defines the attributes and then starts a link to a domain. | 3 |
| LIST | Lists statements in a specific member of the procedure library | 0 |
| LOCK | Suspends current screen processing | 0 |
| LOG | Writes messages to the activity log | 0 |
| LOGSWAP | Swaps logging to the next available log | 3 |
| LSRPOOL | Sets attributes for the VSAM shared resource pool | 2 |
| LTITLE | Sets the title for the Management Services logo panel | 4 |
| LUTRACE | Traces session traffic to selected terminals | 4 |
| MAIDISC | Forces disconnection of an MAI-OC session | 0 |
| MAIINT | Interrupts an MAI-OC application | 0 |
| MAILOGON | Creates an MAI-OC session with another application | 0 |
| MAISEND | Sends data to an application connected by an MAI-OC session | 0 |
| MAISESS | Provides functions for the control of MAI/FS sessions | 0/4 |
| MAXUSERS | Displays and limits the maximum number of concurrent users | 3 |
| MSG | Sends a message to the specified OCS Operator(s) | 0 |
| NCLCHECK | Tests syntax for an NCL procedure without execution | 0 |
| NCLTEST | Sets, resets, or displays NCL test status for this window | 0 |

| Command | Description | Auth |
|---|---|---|
| NCLTRACE | Dynamically alters the trace status of an executing NCL process | 0/2 |
| NDB ALTER | Builds, rebuilds or validates the key indexes for an NDB, or a field | 2 |
| NDB CREATE | Initializes a new NCL database | 2 |
| NDB FIELD | Allows a database field definition to be added, deleted, or updated | 2 |
| NDB PURGE | Frees up a locked or halted database | 2 |
| NDB RESET | Deletes all data from an NCL database | 2 |
| NDB START | Starts an NCL database | 2 |
| NDB STOP | Flags an NCL database as stopping | 2 |
| NDB UNLOAD | Unloads a copy of an NCL database | 2 |
| NETM | Passes a command to NETM for execution | 2 |
| NETMCNTL | Controls communication between Management Services and NEMT | 2 |
| NETSTAT | Invokes the TCP/IP NETSTAT command | T |
| NOTIFY | Sends a message to TSO/TSS users or CMS users | 0 |
| NOTRACE | Terminates a VTAM network trace | 1 |
| NPTAB | Changes the status of a user’s NPF table | 0/4 |
| NRDRET | Restores all Non-Roll Delete messages for an OCS window | 0 |
| NSBRO | Creates and sends a general or specific broadcast | 3 |
| NSLOOKUP | Uses the name service to find the name or address of a host | T |
| NTSDBMOD | Alters database records or session keep counts on the NTS database | 4 |
| NTSMOD | Alters session trace and logging parameters of active sessions | 4 |
| OBEYFILE | Invokes the TCP/IP for MVS OBEYFILE command | T |
| OCSID | Sets or resets OCS window identifiers | 0 |
| OPNDST | Connects a specific terminal to Management Services | 1/0 |

| Command | Description | Auth |
|---|---|---|
| OPSYS | Passes a command to the operating system for execution | 2 |
| ORDER | Reorders your OCS window display from top to bottom | 0 |
| PAGE | Ensures that output messages start at the top of the OCS window | 0 |
| PF | Displays and sets terminal Function keys | 0 |
| PING | Sends echo requests to a remote host | T |
| PPO | Starts and stops the VTAM PPO interface | 3 |
| PROFILE | Displays or modifies a user profile | 0/2 |
| PROFILE EDS | Enables or disables event notification | 0 |
| PURGE | Purges and deletes current timer-initiated commands or NCL locks | 0/2 |
| RECONN | Reconnects the current terminal session with a disconnected region | 0 |
| REPALIAS | Replaces an alias name entry used NEWS | 4 |
| REPCLASS | Replaces an existing SAW, RTM, Session or Resource class definition | 4 |
| REPLINK | Replaces or defines an DLT definition | 3 |
| REPLOGON | Replaces an existing application logon entry | 4 |
| REPLY | Issues a VTAM REPLY command | 1 |
| REPMODE | Replaces or defines an APPC Mode Control Table (MCT) entry | 4 |
| REPOPSET | Replaces or defines an APPC Option Set Control Table (OSCT) entry | 4 |
| REPTRANS | Replaces or defines an APPC Transaction Control Table (TCT) entry | 4 |
| REQMS | Sends data across the CNM interface | 2 |
| RETURN | Exits from any mode or function and returns to the primary menu | 0 |
| ROUTE | Sends a command string to a remote system | 0 |
| SAWARE | Controls NTS activity | 4 |
| SCRIPT | Starts or flushes an MAI-FS script procedure | 0 |

| Command | Description | Auth |
|---|---|---|
| SECUSER | Issues a command to a VS service machine | 2 |
| SHOW AIF | Displays AIF users | 0 |
| SHOW ALLOC | Displays datasets allocated by the ALLOC command | 0 |
| SHOW AOMABEND | Displays diagnostic information if AOM ABENDs | 0 |
| SHOW AOMSTAT | Displays AOM statistics | 0 |
| SHOW APPC | Displays the status of all or selected APPC conversations | 0 |
| SHOW ATF | Displays information about the ATF tables. | 0 |
| SHOW BRO | Displays the current general broadcast messages | 0 |
| SHOW BUFF | Displays the current system storage and buffer utilization | 0 |
| SHOW CNMTRACE | Shows active CNM trace requests | 0 |
| SHOW COMMANDS | Displays commands available to user or system | 0 |
| SHOW COMP | Displays 3270 datastream compression statistics | 0 |
| SHOW CONSOLES | Displays the consoles currently allocated for use by AOM | 0 |
| SHOW DEBUG | Displays NCL debug sessions | 0/2 |
| SHOW DEFALIAS | Displays one or more DEFALIAS entries | 0 |
| SHOW DEFCLASS | Displays NTS class definitions | 0 |
| SHOW DEFLINK | Displays current dynamic link definitions | 0 |
| SHOW DEFLOGON | Displays the current entries in the DEFLOGON table | 0 |
| SHOW DEFMODE | Displays APPC Mode Control Table (MCT) entries | 0 |
| SHOW DEFMSG | Displays the current DEFMSG delivery options | 0 |
| SHOW DEFOPSET | Displays APPC Option Set Control Table (OSCT) entries | 0 |
| SHOW DEFTERM | Displays terminal definitions (VOS3 only) | 0 |
| SHOW DEFTRANS | Displays APPC Transaction Control Table (TCT) entries | 0 |

| Command | Description | Auth |
|---|---|---|
| SHOW DNR | Displays parameters or statistics for the Domain Name Resolver (DNR) | 0 |
| SHOW DOMAINS | Displays the domain information of connected systems | 0 |
| SHOW EDB | Displays information about external database connections | 0 |
| SHOW EDBSTAT | Displays statistics about currently defined EDB connections | 0 |
| SHOW EDBUSER | Displays information about EDB users | 0 |
| SHOW EDBUSTAT | Displays statistics about EDB users | 0 |
| SHOW EDS | Displays current Event Distribution Services PROFILE definitions | 0 |
| SHOW EQUATES | Displays the current EQUATE strings available to this user | 0 |
| SHOW EXEC | Displays the names of members in a procedure library (OS/VS only) | 0 |
| SHOW FTS | Displays the status of file transmissions | 0 |
| SHOW ISR | Displays ISR status information | 0 |
| SHOW ISRSTATS | Displays ISR statistics | 0 |
| SHOW LIB | Displays libraries which have been defined to the system | 0 |
| SHOW LICENSE | Displays the active licensing details and expiry dates | 0 |
| SHOW LINK | Displays the status of INMC or APPC links | 0 |
| SHOW LOCKS | Displays the status of all locks held within the system | 0 |
| SHOW LOGS | Displays the current status of the system activity logs | 0 |
| SHOW LSR | Displays LSR status information | 0 |
| SHOW LUTRACE | Displays the status of all LUTRACE requests | 0 |
| SHOW MAI | Displays information about MAI sessions for this user or all users | 0 |
| SHOW MAISTAT | Displays information about the status of MAI subtasks | 0 |
| SHOW MAP | Displays information about defined Mapping Services maps | 0 |
| SHOW MSGQ | Displays message queue depths | 0 |

| Command | Description | Auth |
|---|---|---|
| SHOW NCL | Displays the status of active NCL processes | 0/2 |
| SHOW NCLGLBL | Displays the names of any defined NCL global variables | 2 |
| SHOW NCLSTAT | Displays the status of NCL procedures currently in storage | 0 |
| SHOW NCLVARS | Displays information on variables in use by NCL procedures | 0/4 |
| SHOW NDB | Displays information about currently active or halted NDBs | 0 |
| SHOW NDBUSER | Displays a list of all users currently signed on to NCL databases | 0 |
| SHOW NETMCNTL | Displays the status of NETM connectivity | 0 |
| SHOW NPF | Displays user's Network Partitioning Facility (NPF) tables | 0 |
| SHOW NPTAB | Displays the status of NPF resource tables | 0 |
| SHOW NRD | Displays the current queue of Non-Roll Delete Messages | 0 |
| SHOW NTS | Displays NTS resource or session information | 0 |
| SHOW NTSDBMOD | Displays currently executing NTSDBMOD & SHOW SKEEP processes | 0 |
| SHOW NTSSTATS | Displays NTS statistics | 0 |
| SHOW NTSUSER | Displays NTS users | 0 |
| SHOW OCS | Displays information about OCS users | 0 |
| SHOW PANELS | Displays Management Services panel queue information | 0 |
| SHOW PARM | Displays Management Services startup parameters | 0 |
| SHOW PATH | Displays current panel library path definitions | 0 |
| SHOW PAUSE | Displays the current PAUSE or WAIT status | 0/2 |
| SHOW PPIUSERS | Displays PPI user statistics | 0 |
| SHOW PPOSTAT | Displays PPO/SPO statistics | 0 |
| SHOW REPLY | Displays VTAM messages that require a reply | 0 |
| SHOW SCNT | Displays the current session count | 0 |

| Command | Description | Auth |
|---|---|---|
| SHOW SERVER | Displays the status of registered server processes | 0 |
| SHOW SESS | Displays the terminals that are in session | 0 |
| SHOW SKEEP | Displays the NTS session keep counts for historical sessions | 0 |
| SHOW SNAMS | Displays SNA Management Services information | 0 |
| SHOW SOCKETS | Displays information about the use of TCP/IP services | T |
| SHOW SSISTATS | Displays Subsystem Interface (SSI) statistics | 0 |
| SHOW SSIUSERS | Displays information about SSI users | 0 |
| SHOW STRACE | Displays NTS session trace activity | 0 |
| SHOW SUBSYS | Displays user subsystem status | 0 |
| SHOW SYSCONS | Displays a list of currently logged on operating system consoles | 0 |
| SHOW SYSPARMS | Displays the current SYSPARMS settings | 0 |
| SHOW TCPIP | Displays information about the use of TCP/IP services | T |
| SHOW TERM | Displays the terminals that are in session | 0 |
| SHOW TIMER | Displays current timer initiated commands | 0 |
| SHOW TIMEZONE | Displays current time zones and the system time zone offset | 0 |
| SHOW TSO | Displays current TSO users | 0 |
| SHOW TSS | Displays current TSS users | 0 |
| SHOW UDB | Displays VSAM dataset information | 0 |
| SHOW UDBUSER | Displays NCL UDB user information | 0 |
| SHOW USERS | Displays the current signed on users | 0 |
| SHOW VARTABLES | Displays information about VARTABLES | 0 |
| SHOW VMOP | Displays VMOPERATOR active sessions | 0 |
| SHOW VSAM | Displays extended VSAM dataset information | 0 |

| Command | Description | Auth |
|---|---|---|
| SHOW XMIT | Displays the status of all or selected FTS Transmission Requests | 0 |
| SHOW XNFTRACE | Displays the status of all XNFTRACE requests | 0 |
| SHUTDOWN | Commences or cancels an orderly shutdown of the system | 3 |
| SIGNOFF | Signs off from a remote system | 0 |
| SIGNON | Signs on to a remote Management Services system | 0/2 |
| SPLIT | Opens or adjusts size of an OCS window | 0 |
| SPO | Starts and stops the VTAM Secondary Program Operator interface | 3 |
| SSI | Signs off or stops Subsystem Interface (SSI) | 0/3 |
| START | Starts executing an asynchronous NCL process | 0 |
| STATUS | Displays current general system status | 0 |
| STRACE | Starts and stops session trace activity | 4 |
| SUBMIT | Passes a command to a background environment for processing | 0/2 |
| SUBSYS | Defines, starts, stops and deletes subsystems | 3 |
| SUSLOGON | Suspends an entry or entries in the DEFLOGON table | 3 |
| SWAP | Swaps current logical windows | 0 |
| SYSCMD | Sends a command to the operating system | 2 |
| SYSLOG | Issues VMOPERATOR log browse commands | 2 |
| SYSMON | Sends data to System Monitor in 3600/4700 | 1 |
| SYSPARMS | Defines or changes system default values. This command is described in Appendix D, *SYSPARMS Operands*, in this manual. | 4/9 |
| TERMINAL | Provides extended color and/or highlighting datastreams to terminals | 0 |
| TCPIP MODIFY | Alters the current trace options setting. | 4 |
| TCPIP QUIESCE | Stops the interface when all the sockets are closed. | 4 |
| TCPIP START | Initiates TCP/IP services. | 4 |

| Command | Description | Auth |
|---|---|---|
| TCPIP STOP | Terminates TCP/IP services. | 4 |
| TELNET | Starts a Telnet connection | T |
| TIME | Sets and displays the current date and time | 0 |
| TIMEZONE | Maintains time zone names and offsets | 3 |
| TITLE | Sets the title to be displayed on the top line of the OCS window | 4 |
| TNCMD | Sends a Telnet command on a connection to a remote host | T |
| TNDISC | Disconnects a Telnet connection | T |
| TNSEND | Sends data on a Telnet connection | T |
| TRACE | Initiates a VTAM network trace | 1 |
| TRACEROUTE | Traces the route taken by TCP/IP packets to a remote host | T |
| TRANSMIT | Requests a dataset transmission | 0 |
| UDBCTL | Controls the status of User DataBases (UDBs) | 3/4 |
| UNALLOC | Unallocates a closed dataset | 0/3 |
| V | Issues a VTAM VARY command | 1 |
| X | Exits from OCS screen mode | 0 |
| XLATE | Tests alias name translation | 1 |
| XNF | Stops all XNF connections | 3 |
| XNFTRACE | Initiates or terminates tracing of XNF connectivity | 4 |

# Dataset Authorization Exits Support

This appendix describes how to write and install the dataset access authorization exit and the dataset services authorization exit.

## Writing a Dataset Access Authorization Exit

The authorization exit is attached and operates as a subtask. Therefore, complex processing, WAITs, or SVCs do not impact overall Management Services performance.

### Registers on Entry to the Exit

When the exit is invoked, Register 1 contains the address of a communication area containing various parameters. This communication area can be mapped using the macro $NMDSNCK, supplied in the distribution libraries. This macro provides a DSECT expansion to perform the mapping, and detailed information on 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 content on entry, with the exception of Register 15 which should contain a return code.

## Parameters Passed to NMDSNCHK

NMDSNCHK is provided so that the installation can determine whether access to a dataset is to be permitted. When invoked as a result of an ALLOCATE command, a communication area containing the following information is passed to the exit:

- Dataset name
- Management Services user ID of requestor

The format of this area is mapped by the supplied $NMDSNCK DSECT.

### Identifying the Type of Call

The type of call being made to the exit is identified by the DSNSTYPE field in the exit communication area. This field is set by Management Services as follows:

| Value | Meaning |
|---|---|
| F'0' | FTS call |
| F'4' | FTS call |
| F'8' | FTS call |
| F'12' | FTS call |
| F'16' | Called for an ALLOCATE command |

If your installation uses RACF security software, the exit can make a call to RACF to associate the allocation with the requesting user ID rather than with Management Services itself.

## Return Codes From the Exit

A return code is set in Register 15 on return from all calls to the exit, indicating the action that should be taken:

| Return Code | Meaning |
|---|---|
| 0 | Access permitted. Allocation can proceed. |
| 4 | Access denied. An error message can be placed in the field DSNCTMSG, and its length in the field DSNCLMSG, in the communication area. |

## Installing the Dataset Access Authorization Exit

The installation of the dataset access authorization exit is different depending on the type of operating system you have. See the relevant section below for information on how to install the exit in your operating system.

### Installing the Exit in OS/390, z/OS, and MSP Systems

In OS/390, z/OS, and MSP systems, the exit is installed with the distributed product under the control of SMP. If a new exit is to be installed, replacing the default exit, it must be applied as an SMP usermod.

The following SMP control statements can be used as a guide:

```
++USERMOD(usermod).
++VER(Z038) FMID(NMB0000).
++SRC(NMDSNCHK) DISTLIB(INSTALL) TXLIB(userlib).
```

**Note**

When an installation-written exit is implemented in this manner it is advisable that you check the SMP regression reports following the application of maintenance or new product releases to ensure that the exit is not regressed.

### Installing the Exit in VOS3 Systems

In VOS3 systems, the exit is distributed in the Management Services load library. The source is distributed in the BASE.INSTALL library. If a new exit is to be installed, replacing the default exit, it must be assembled and linked as NMDSNCHK to a load library allocated to the job or started task.

**Note**

When NMDSNCHK is replaced with an installation-written module, it is necessary to reinstall the installation version when new releases of the product are installed.

### Installing the Exit in VM Systems

In VM systems, the exit is distributed as a member of the LOADLIB, and the sample source as NMDSNCHK ASSEMBLE, both on the installation minidisk. If a new exit is to be installed, replacing the default exit, it must be assembled and linked as NMDSNCHK to a LOADLIB that is accessible to the Management Services GCS user ID. Management Services links to this load module when the exit is driven.

**Note**

When NMDSNCHK is replaced with an installation-written module, it is necessary to reinstall the installation version when new releases of the product are installed.

## Writing a Dataset Services Authorization Exit

The NMDSSCHK exit is loaded by a Dataset Services subtask and is called synchronously by the subtask. The exit can issue WAIT or other SVCs without impacting the Management Services main task.

The exit load module must be named NMDSSCHK and be link edited as AMODE 24, RMODE 24. The exit procedure can be coded as reentrant; however, this is not required because an explicit load is performed for each task using the exit. The exit is always called in 24-bit address mode.

### Exit Environment

The exit is loaded separately by each invocation of the Data Services subtask. For example, a subtask is created to allocate or free a dataset, or to open a dataset, and so on.

The exit receives (per subtask invocation) one initialization call, possibly one or more logical function calls, and possibly one termination call. For example, copying a dataset from A to B would result in the logical calls shown below:

```
ALLOC A    task init
ALLOC A    ALLOC call
ALLOC A    task terminate
ALLOC B    task init
ALLOC B    ALLOC call
ALLOC B    task terminate
READ A     task init
READ A     open check
WRITE B    task init
WRITE B    open check
READ A     task terminate
WRITE B    task terminate
```

```
FREE A     task init
FREE A     ALLOC call
FREE A     task terminate
FREE B     task init
FREE B     ALLOC call
FREE B     task terminate
```

If the exit is defined as reentrant you can use the DSSCXCOR field to anchor any private storage required by the subtask.

**Note**

Do not use shared (SP 50) storage as an exit work area. See additional notes in the section, *Exit Termination Call*, on page I-7.

## Registers on Entry to the Exit

On entry to the exit, Register 1 contains the address of a parameter list area that is used to pass information to the exit. This communication area is mapped by the $NMDSSCK macro, that is distributed with Management Services. This macro provides a DSECT expansion to perform mapping and contains detailed information on the contents of each field.

Standard module linkage conventions apply; on entry, the exit must save the contents of all registers (Register 13 contains the address of a save area that can be used by the exit) and on exit all registers must be restored to the value they had on entry with the exception of Register 15 that should contain a return code relevant to the function call.

### Exit Initialization Call

This call allows the exit to perform any initialization required. This could include authorizing the subtask against the authority level of the user. This call is not specific to any request from the user: logical function calls identify the functions that have been requested by a user and you can perform more specific processing for these calls.

The parameter area passed to the exit contains the user ID of the user, any User Token or Security Exit Correlator associated with the user (drawn from the main Management Services security exit, if one is being used) and information about the Management Services system in which the exit is running (for example, the NMID).

The exit must return with Register 15 set to a return code that indicates the success or failure of exit initialization. A return code can be set to indicate to Management Services that no more security processing is required for the function.

The following return codes are allowed for this call:

| Code | Meaning |
|---|---|
| 0 | Exit initialization completed successfully |
| 4 | Exit initialization completed successfully. No more logical function calls are required for this user—note that the termination call is still performed, however. See *Using Return Code 4,* below, for details |
| 8 | Exit initialization failed. |

If Register 15 is set to 8, you can use the DSSCLMSG and DSSCTMSG fields to return a message indicating the reason for the failure. See the $NMDSSCK macro for full details. If no message is returned, Dataset Services sets a default message. The message returned by the exit is recorded on the activity log.

### *Using Return Code 4*

If you set return code 4 on initialization this means that the host does not need to participate on any subsequent logical function authorization.

You might do this, for example, if you issue a RACINIT during initialization to register the subtask as relating to a specific user ID (that is, when you have a security exit installed). In this case, for example, an OPEN DATASET call would result in an S913 OPEN ABEND since RACF does the security checking. Note that Dataset Services handles this situation correctly.

If you set return code 0, but still connect the subtask to the security system, you can do additional checking on some calls (for example, rename PDS member) that the security system does not specifically check.

## Logical Function Call

This call represents a function that has been requested by the user. For example, a dataset allocation request or a request to open a dataset. The DSSCTYPE field is used to indicate the function to be performed.

The parameter area passed to the exit contains all information passed on the exit initialization call as well as any specific information requested by the user for the function. For example, if the call is for an allocation request the dataset name, disposition, volume name, and so on, might be available. The $NMDSSCK macro contains a full description of each logical function call and the parameters that are passed on each call.

The exit can return with Register 15 set to a return code that indicates whether the function is permitted.

The following return codes are allowed for this call:

| Code | Meaning |
|---|---|
| 0 | Function is authorized |
| 4 | Function is not authorized |

If Register 15 is set to 4, a message indicating the reason for the failure can be returned using the DSSCLMSG and DSSCTMSG fields. See the $NMDSSCK macro for full details. If no message is returned by the exit, Dataset Services sets a default message. The message returned by the exit is recorded on the activity log.

### Exit Termination Call

This call allows the exit to clean up any allocated storage areas and, if applicable, break the connection with any security subsystem that is being used. The exit load module is deleted from storage on return from this call.

**Note**

This call may not occur. If, for example, an NCL process using Dataset Services is flushed, the subtask(s) are force detached. You should be aware of this and allocate any private storage in a non-shared subpool (this can include subpool SP0 but not SP50), which results in the storage being automatically freed.

The parameter passed to the exit contains all information passed on the exit initialization call. Register 15 must contain 0 on return from this call.

## Installing the Dataset Services Authorization Exit

In OS/390, z/OS, and MSP systems, the exit is installed with the distributed product under the control of SMP. If a new exit is to be installed, replacing the default exit, it must be applied as an SMP usermod.

The following SMP control statements can be used as a guide:

```
++USERMOD( usermod).
++VER(Z038) FMID(NMC20MS).
++SRC(NMDSSCHK) DISTLIB(INSTALL) TXLIB(userlib).
```

**Note**

When an installation written exit is implemented in this manner it is advised that SMP regression reports be checked following the application of maintenance or new product releases to ensure that the exit is not regressed.

# J

# INMC Security Exit Support

This appendix describes how to code the INMC Security Exit and details the parameter lists exchanged between Management Services and the exit. For more information about the INMC exit, see Chapter 4, *Implementing Additional Security*.

## Writing an INMC Security Exit

You need to write a primary and secondary exit for each system that is to be controlled by the INMC security exit when connected by an INMC link.

The exits must be assembled and linked to form an executable load module and must be placed in a load library, or a core image library accessible to Management Services during execution.

When the exit is called, standard assembler language linkage conventions apply.

When invoked, the exit operates under the Management Services main task, and therefore extensive processing, I/O operations, or WAITs issued within the exit can impact the overall performance of the system.

## Identifying the Primary Exit to Management Services

One INMC primary exit is available for any one system and must be identified to Management Services by using the following command:

```
SYSPARMS INMCEX01=exitname
```

where *exitname* is the load module or phase name of the primary exit. This command must be included in the RMINIT initialization parameters or issued as a command before a link is activated.

When an INMC link is activated, INMC attempts to load a copy of the exit. If the load fails for any reason it is regarded as a security exposure and the link is disabled automatically. The exit should be written to be reentrant if possible since a load is issued for each link that is activated.

## Identifying the Secondary Exit to Management Services

One INMC secondary exit is available for any one system and must be identified to Management Services by the command:

```
SYSPARMS INMCEX02=exitname
```

where *exitname* is the load module or phase name of the secondary exit. This command must be included in the RMINIT initialization parameters or issued as a command before a link is activated.

When an INMC link is activated, INMC attempts to load a copy of the exit. If the load fails for any reason the primary exit of the remote system is notified that no secondary exit exists in this system. In this case the remote primary exit decides whether to allow link activation.

## Changing Exit Names Dynamically

The name of the load module that forms either the primary or secondary INMC exit can be changed dynamically at any time by reissuing the SYSPARMS command to identify the new exit name. Either exit can be disabled at any time by issuing the following commands:

```
SYSPARMS INMCEX01=NONE
SYSPARMS INMCEX02=NONE
```

### Registers on Entering INMC Exits

Both primary and secondary exits are called using conventional linkage. The registers on entry to either exit contain values as follows:

Register 0 : Unpredictable
Register 1 : Address of parameter list
Register 2 : Unpredictable
Register 3 : Unpredictable
Register 4 : Unpredictable
Register 5 : Unpredictable
Register 6 : Unpredictable
Register 7 : Unpredictable
Register 8 : Unpredictable
Register 9 : Unpredictable
Register 10 : Unpredictable
Register 11 : Unpredictable
Register 12 : Unpredictable
Register 13 : Address of standard save area
Register 14 : Management Services return address
Register 15 : Entry point of INMC primary or secondary exit

The exit must save registers on entry and perform standard save area linkage. On return, registers must be restored and control returned to the address held in Register 14 on entry.

## Writing a Primary Exit

The primary exit is called by Management Services with Register 1 containing the address of a parameter list which is always ten consecutive full words in length.

Depending upon the reason for the call, some of these words might be set to binary zeroes.

The first word of the parameter list is always the address of a full word that contains a function code identifying the type of call being made.

The other parameters passed depend upon the value of this function code and the exit must therefore determine the function code first in order to decide which parameters to expect.

The parameter list passed to the exit is also used as a parameter list returned from the exit, allowing the exit to indicate the processing required and to pass the appropriate information to Management Services.

**Note**

On a call to the exit, addresses of various fields are included in the parameter list. Only these fields can be used to return information from the exit; the exit cannot pass parameters back to Management Services in any other locations.

## Specifying Initialization Processing

Initialization processing notifies the primary exit of two things:

- The INMC link to the specified remote system has been activated
- Whether the remote system is configured with a secondary exit to which the primary exit can send messages

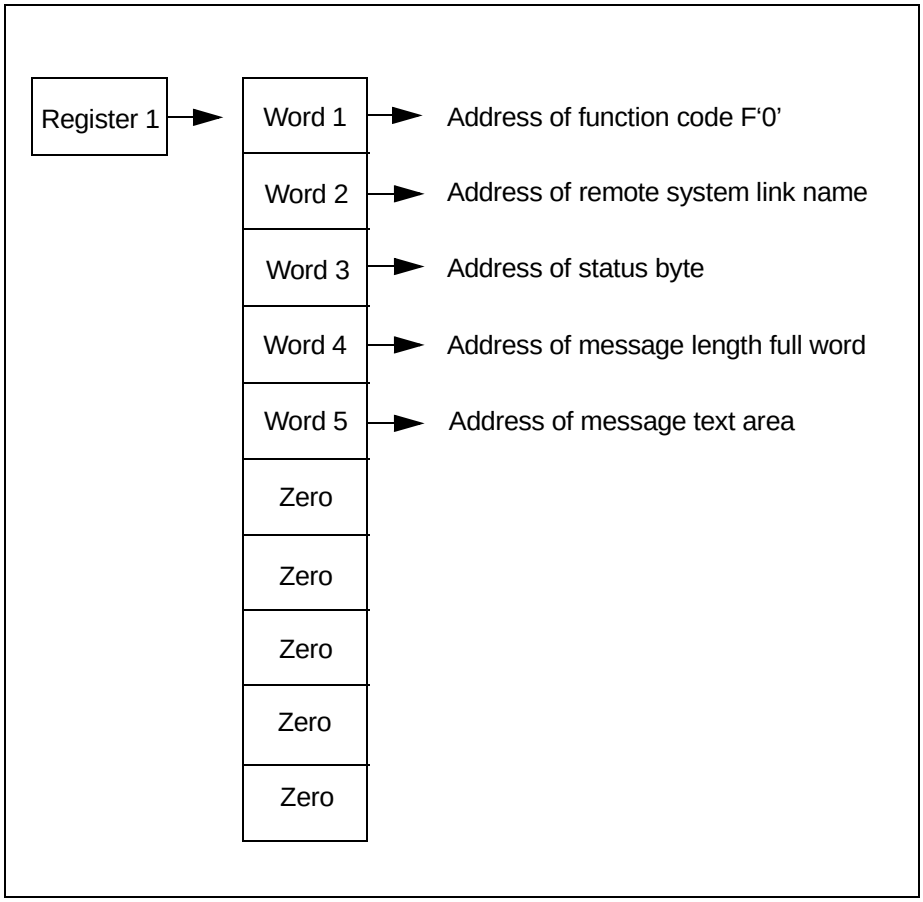
To provide this functionality in your exit you need to code the initialization call to correspond with the parameter list below.

You need to take into consideration any special processing you want to perform, such as, password exchange between the connecting systems.

### Initialization Call Parameter List

On entry to the Primary exit Register 1 points to a list of ten full words, as shown in Figure J-1.

*Figure J-1. Primary Exit Initialization Call Parameter List*




**Word 1** of the parameter list points to a full word containing function code X‘00000000’. Other words in the parameter list point to additional parameters as follows:

**Word 2**

The address of the 1 to 12 character link name of the remote system with which a link has been established.

**Word 3**

The address of a 1-byte field that specifies the status of the secondary exit in the remote system. Values of this field and the meanings are as follows:

**X‘00’**

A secondary exit is available for communications.

**X‘04’**

The remote system does not have a secondary exit defined. No SYSPARMS INMCEX02 command has been issued in the remote system.

**X‘08’**

The remote system has a secondary exit defined but the load module (or phase) was not loaded successfully.

**Word 4**

The address of a full word, value F‘0’. The primary exit can place in this field the length of a message that is to be sent to the secondary exit in the remote system. The maximum length is 256 bytes (decimal). If a longer length is specified it is truncated to 256. If a negative length is specified it is forced to 256.

**Word 5**

The address of an area in which the primary exit can place the text of a message that is to be sent to the secondary exit in the remote system. The message is assumed to be left aligned in this message area. The maximum message text length is 256 (decimal) bytes. The data in the message can be binary or character and is transparent to Management Services. The meaning and format of the message sent to the secondary exit is determined by the installation.

**Words 6 to 10**

Set to zero.

## Return Codes from Initialization Call

On return to Management Services from the Initialization Call, the primary exit signals what it wants done next. The choices that are available are:

- Send a message to the secondary exit in the remote system
- Terminate exit processing

The exit indicates which option is required by using the same ten full words of the parameter list with which it was called, to pass back to Management Services its own instructions. When the exit returns to Management Services therefore, the ten full word parameter list must be formatted as follows:

**Word 1**

The address of a function code specifying the action that the exit wishes Management Services to take. Values of this function code and their meanings are as follows:

**F‘0’**

A message is to be sent to the secondary exit in the remote system.

**F‘4’**

Exit processing is to terminate.

**Word 2**

Unchanged and not used.

**Word 3**

The address of a 1-byte field. This is the same address as was passed to the exit in word 3 of the parameter list. The value of this 1-byte field depends upon the function code returned by the exit in word 1. Valid values and their meanings are as follows:

**F‘0’**

Function code not used.

**F‘4’**

X‘00’ Allow normal link activation

X‘04’ Allow link to remain open but allow no traffic to flow

X‘08’ Close the link

**Word 4**

This word is not used for function code F‘4’. For function code F‘0’ this word holds the address of a full word in which the primary exit can place the binary length of a message that is to be sent to the secondary exit in the remote system. The maximum length is 256 bytes (decimal). If a longer length is specified it is truncated to 256. If a negative length is specified it is forced to 256. This is the same address as passed to the exit in word 4 of the parameter list.

**Word 5**

This word is not used for function code F‘4’. For function code F‘0’ this word holds the address of an area in which the primary exit can place the text of a message that is to be sent to the secondary exit in the remote system. The message is assumed to be left aligned in this message area. The maximum message text length is 256 (decimal) bytes. The data in the message can be binary or character and is transparent to Management Services. The meaning and format of the message sent to the secondary exit is determined by the installation. This is the same address as passed to the exit in word 5 of the parameter list.

**Words 6 to 10**

Set to zero.

## Specifying Message Delivery Processing

When the primary exit sends a message to the secondary exit in a remote system, the secondary exit always responds with a reply. That reply can be a null message of zero length. The reply message is presented to the primary exit for processing by using a Deliver Call.

The primary exit performs whatever processing is required and then returns to Management Services with the ten full word parameter list set to indicate the processing required next.

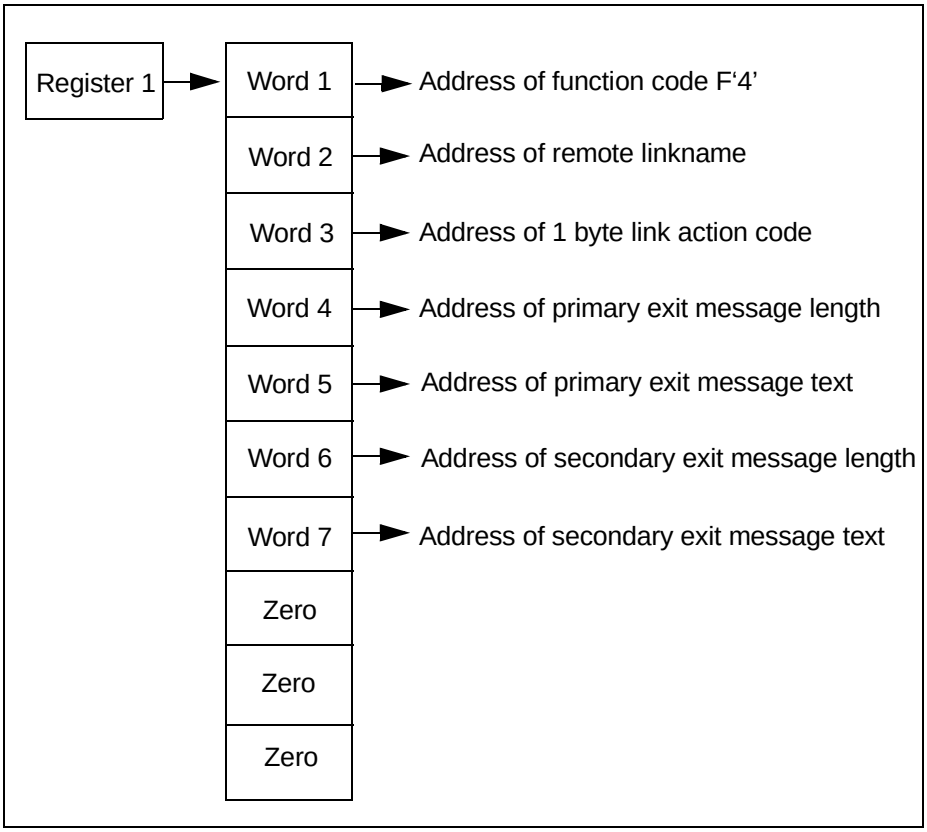
On return to Management Services the exit can request one of two options:

- Exit processing is to terminate
- A message is to be sent to the secondary exit

### Delivery Call Parameter List

On entry to the primary exit, Register 1 points to a list of ten full words, as shown in Figure J-2.

*Figure J-2. Primary Exit Delivery Call Parameter List*




**Word 1** of the parameter list points to a full word containing function code X‘00000004’. The other words are set as follows:

**Word 2**

The address of the link name of the remote system with which a link has been established.

**Word 3**

The address of a full word in which the primary exit can return a code indicating whether the link is to be opened for normal INMC traffic.

**Word 4**

The address of a full word, value F‘0’. The primary exit can place in this field the length of a message that is to be sent to the secondary exit in the remote system. The maximum length is 256 bytes (decimal). If a longer length is specified it is truncated to 256. If a negative length is specified it is forced to 256.

**Word 5**

The address of an area in which the primary exit can place the text of a message that is to be sent to the secondary exit in the remote system. The message is assumed to be left aligned in this message area. The maximum message text length is 256 bytes (decimal). The data in the message can be binary or character and is transparent to Management Services. The meaning and format of the message sent to the secondary exit is determined by the installation.

**Word 6**

The address of a full word containing the length of a message sent from the secondary exit in the remote system. The maximum length is 256 bytes (decimal), minimum is zero (which would be a null message).

**Word 7**

The address of the message sent from the secondary exit.

**Words 8 to 10**

Set to zero.

## Return Codes from Delivery Call

When the primary exit completes its processing of the message returned from the secondary exit it formats the ten word parameter list to indicate the processing required next and then returns to Management Services.

The parameter list must be set exactly as described in the section, *Return Codes from Initialization Call*, on page J-6.

## Specifying Termination of Link Notification Processing

If a link to a remote system is lost before a reply is received from the secondary exit, Management Services indicates this to the primary exit via a notify call. This call tells the primary exit that no further communications can be received from the secondary exit and that the INMC link to the remote system has been lost.

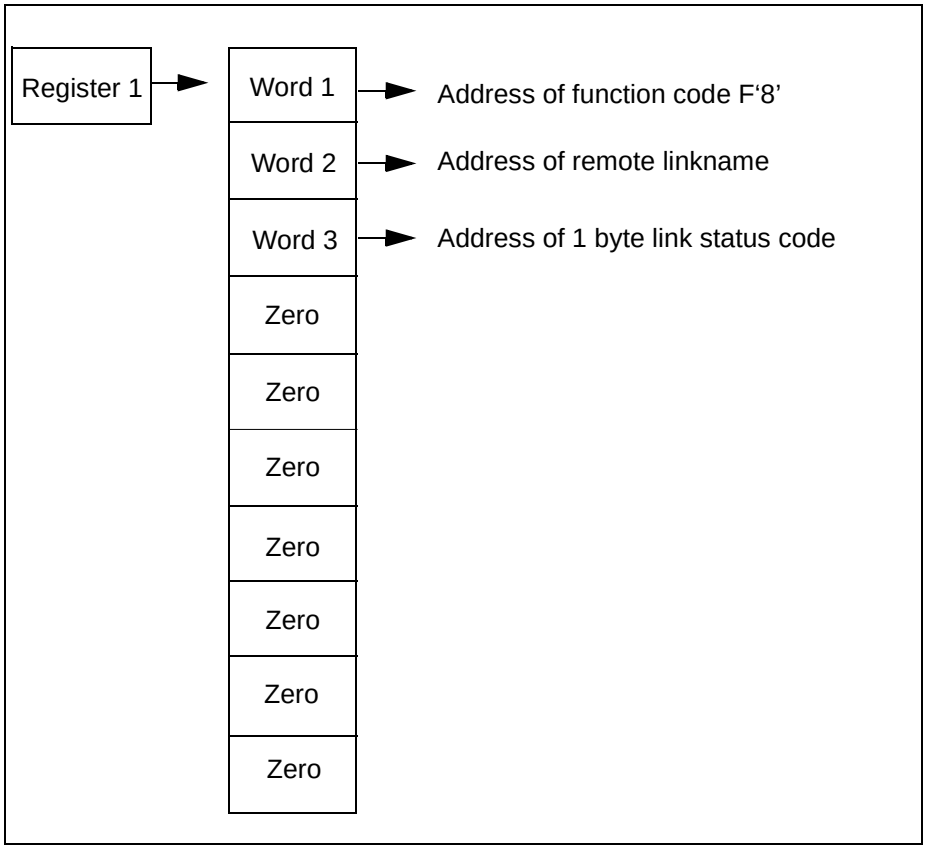
A notify call is also made in response to the primary exit attempting to communicate with a secondary exit when the remote system is not configured with a secondary exit or the secondary exit failed to initialize.

When the exit has been notified of a lost link, it should then perform whatever termination processing it needs to do then return to Management Services, indicating that exit processing is to terminate. Any other action results in repetitive notify calls until the exit signals that exit processing is to end.

### Notify Call Parameter List

On entry to the primary exit, Register 1 points to a list of ten full words, as shown in Figure J-3.

*Figure J-3. Primary Exit Notify Call Parameter List*




**Word 1** of the parameter list points to a full word containing function code X‘00000008’. The other words and their meanings are as follows:

**Word 2**

The address of the link name of the remote system with which contact has been lost.

**Word 3**

The address of 1-byte field containing a notify code. Valid values are as follows:

**X‘04’**

The remote system does not have a secondary exit defined. No SYSPARMS INMCEX02 command has been issued in the remote system.

**X‘08’**

The remote system has a secondary exit defined but the load module (or phase) was not loaded successfully.

**X‘0C’**

The link has been lost.

**Words 4 to 10**

Set to zero.

## Return Codes from Notify Call

When the primary exit completes its processing of the notify call it should format the ten word parameter list as follows:

**Word 1**

Address of a function code specifying the action that the exit wishes Management Services to take. Values of this function code must be:

**F‘4’**

Exit processing is to terminate.

The remaining nine full words should be returned unchanged.

## Writing a Secondary Exit

The secondary exit is called by Management Services with Register 1 containing the address of a parameter list which is always ten consecutive full words in length.

Depending upon the reason for the call, some of these words can be set to binary zeroes.

The first word of the secondary exit parameter list is always the address of a full word that contains a function code identifying the type of call being made.

The other parameters passed depend upon the value of this function code, and the exit must therefore determine the function code first in order to decide which parameters to expect.

The parameter list passed to the exit is also used as a parameter returned from the exit, allowing the exit to indicate the processing required and to pass the appropriate information to Management Services.

**Note**

On a call to the exit, addresses are included in the parameter list of various fields. Only these fields can be used to return information from the exit; the exit cannot pass parameters back to Management Services in any other locations.

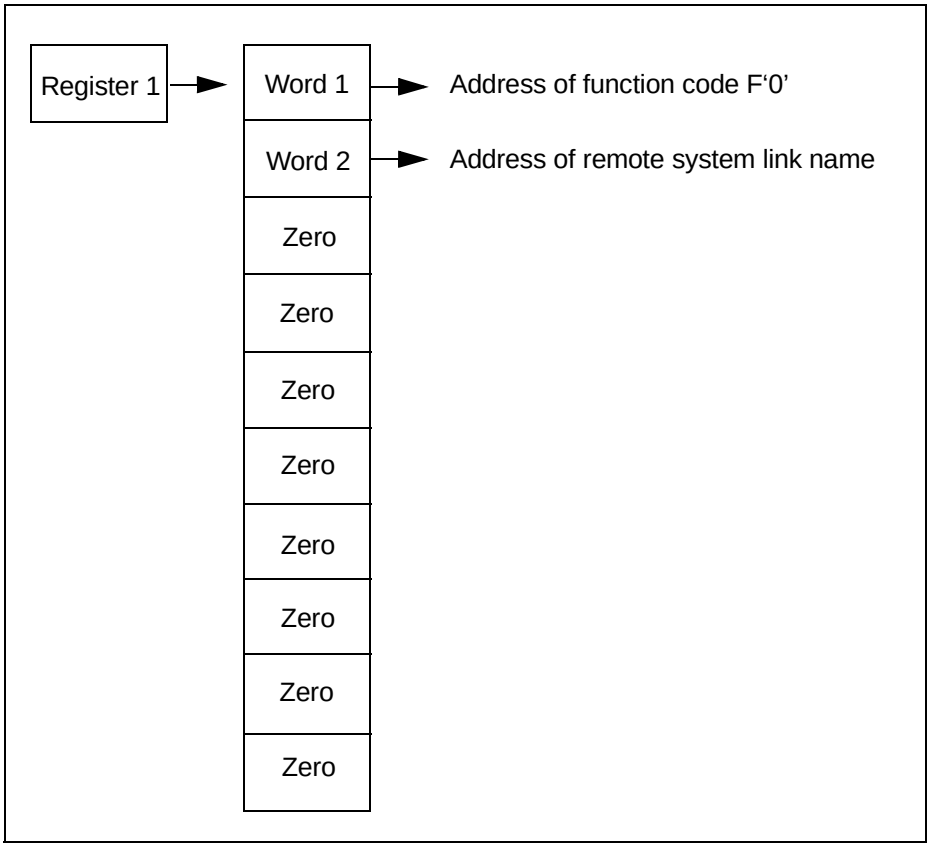
### Specifying Initialization Processing

Initialization processing indicates to the secondary exit that a link has become active. There is no indication whether a primary exit exists in the remote system, or, if there is, whether that exit is going to attempt to communicate with the secondary exit. The secondary exit must therefore be written to expect whatever processing is implemented in the remote primary exit.

#### Initialization Call Parameter List

On entry to the secondary exit Register 1 points to a list of ten full words, as shown in Figure J-4.

*Figure J-4. Secondary Exit Initialization Call Parameter List*




**Word 1** of the parameter list points to a full word containing function code X‘00000000’. Other words in the parameter list point to additional parameters as follows:

**Word 2**

The address of the link name of the remote system with which a link has been established. There may or may not be a primary exit defined for that system. If there is, it is up to the primary exit to determine whether it wants to communicate with this secondary exit.

**Words 3 to 10**

Set to zero.

## Return Codes from the Initialize Call

The secondary exit does not have an opportunity to respond to the initialize call. On completion of any processing it chooses to perform when passed control, the secondary exit should return to Management Services with its ten full word parameter list unchanged.

## Specifying Message Delivery Processing

When the primary exit sends a message to the secondary exit the message is presented to the secondary exit for processing using the Deliver Call.

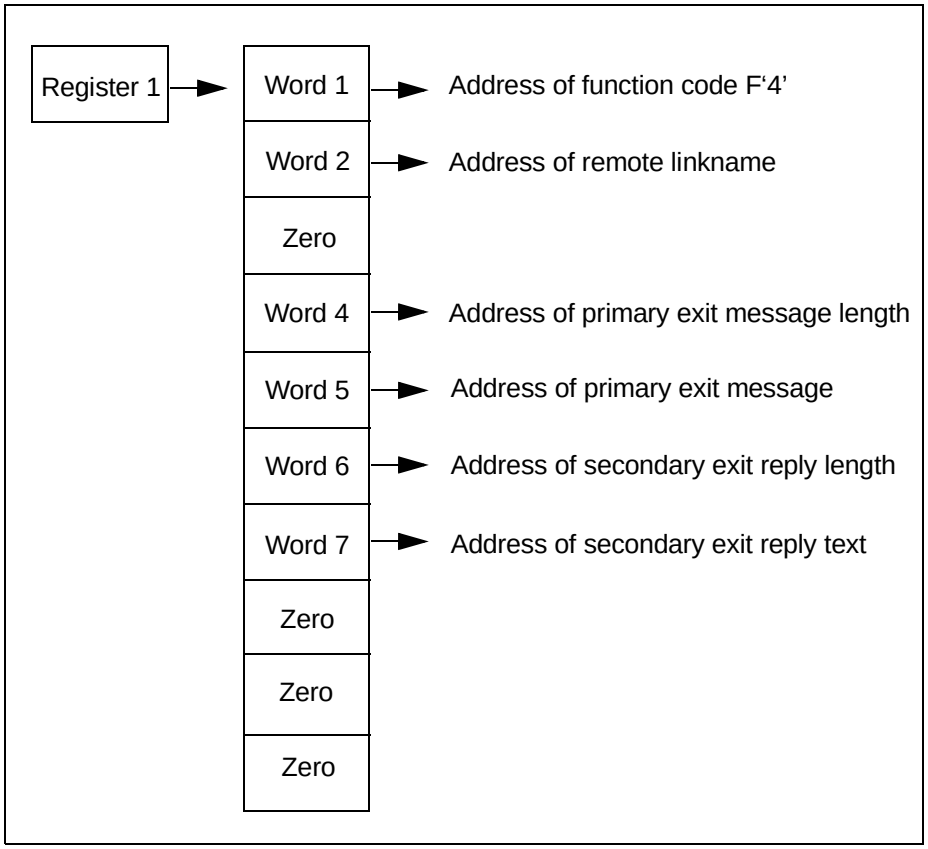
The secondary exit performs whatever processing is required and then returns to Management Services. On return, Management Services expects the secondary exit to have set the message-length and message-text areas pointed to by words 6 and 7 of the parameter list with which it was called.

The secondary exit can, if necessary, indicate that a null message is to be returned to the primary exit, perhaps as an acknowledgment to the message sent by the primary.

### Deliver Call Parameter List

On entry to the secondary exit, Register 1 points to a list of ten full words, as shown in Figure J-5.

*Figure J-5. Secondary Exit Deliver Call Parameter List*




**Word 1** of the parameter list points to a full word containing function code X‘00000004’. The other words are set as follows:

**Word 2**

The address of the link name of the remote system from which the message has been received.

**Word 3**

Set to zero.

**Word 4**

The address of a full word containing the length of a message sent from the primary exit in the remote system. The maximum length is 256 bytes (decimal), minimum is zero (which would be a null message).

**Word 5**

The address of the message sent from the primary exit.

**Word 6**

The address of a full word, value F‘0’. The secondary exit can place in this field the length of a message that is to be sent to the primary exit in the remote system. The maximum length is 256 bytes (decimal). If a longer length is specified it is truncated to 256. If a negative length is specified it is forced to 256.

**Word 7**

The address of an area in which the secondary exit can place the text of a message that is to be sent to the primary exit in the remote system. The message is assumed to be left aligned in this message area. The maximum message text length is 256 bytes (decimal). The data in the message can be binary or character and is transparent to Management Services.

The meaning and format of the message sent to the primary exit is determined by the installation.

**Words 8 to 10**

Set to zero.

### Return Codes from Secondary Exit Deliver Call

The secondary exit should set the full word pointed to by word 6 of the call parameter list to the length of the message text that is to be returned to the primary exit as a reply to the message just delivered.

The message length has a range of 0 to 256 in hexadecimal. Any text outside the specified range is truncated to 256 bytes decimal. A zero length is accepted. Negative length settings are forced to a zero value.

Word 7 of the call parameter list points to a 256-byte area in which the secondary exit can place the text of the message to be returned to the primary exit. The format and content of the message returned is decided by the installation.

No other return information is accepted from the secondary exit.

## Specifying Termination Processing

When the primary exit in a remote system decides that the primary/secondary exit exchange of information is complete it returns to Management Services indicating that exit processing is to terminate and also specifies whether the link is to be activated or closed down.

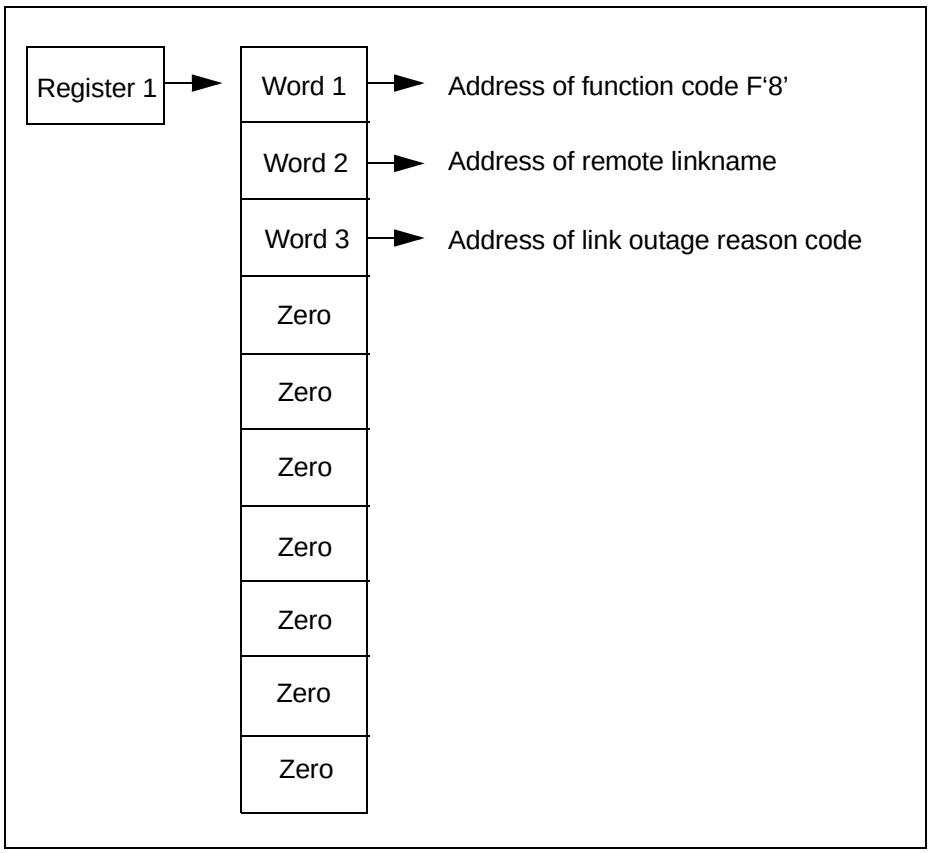
The decision to finish communication is reported to the secondary exit as a disconnect call. This allows the secondary exit to determine that no further messages are to be received from the primary exit and to release any exit dependent resources it acquired.

The disconnect call is also issued when the link to the primary exit's remote system is lost.

## Disconnect Call Parameter List

On entry to the secondary exit, Register 1 points to a list of ten full words, as shown in Figure J-6.

*Figure J-6. Secondary Exit Disconnect Call Parameter List*




**Word 1** of the parameter list points to a full word containing function code X'00000008'. The other words are set as follows:

**Word 2**

The address of the link name of the remote system from which the message has been received.

**Word 3**

The address of 1 byte disconnect reason code. Valid values are as follows:

**X'00'**

Orderly termination of communication by choice of the primary exit.

**X‘04’**
Disconnection caused by loss of contact.

**Words 4 to 10**
Set to zero.

## Return Codes from the Disconnect Call

The secondary exit should clean up any resources allocated during the conversation with the primary exit and perform any required termination processing. No further communication is possible with the primary exit.

No parameters can be returned to Management Services.

On return to management service, the link either is out of action (if caused by a link outage) or is activated or deactivated according to the choice of the primary exit in the remote system.

# K

# The NCL Authorization Exit Communication Area

This appendix provides an example of the NCL authorization exit communication area.

## Communication Area

| Offsets | Name | Length | Format | Description |
|---|---|---|---|---|
| 0 ( 0) | NEXFUNC | 4 | HEX | FUNCTION CODE<br>X‘00000000’ = &FILE OPEN VALIDATION REQUEST.<br>X‘00000004’ = &SMFWRITE AUTHORIZATION REQUEST.<br>X‘00000008’ = &EDB, &SQL REQUESTS<br>Y‘0000000C’ = &NDBOPEN REQUEST |
| 4 ( 4) | NEXCORR | 4 | HEX | LOGON SECURITY EXIT CORRELATOR,<br>IF SET BY INSTALLATION SECURITY EXIT. |
| 8 ( 8) | NEXUTOKEN | 4 | HEX | LOGON SECURITY EXIT USER TOKEN,<br>IF PROVIDED BY INSTALLATION SECURITY EXIT. |
| 12 ( C) | | 4 | HEX | RESERVED |
| 16 (10) | NEXUSER | 8 | CHARACTER | REQUESTING USERID, OR PSUEDO-USERID<br>IF SYSTEM ENVIRONMENT. |
| 24 (18) | NEXLUNAM | 8 | CHARACTER | LUNAME OR REGION NAME |
| 32 (20) | NEXPRCO | 8 | CHARACTER | &0 NCL PROCEDURE. |

| Offsets | Name | Length | Format | Description |
|---|---|---|---|---|
| 40 (28) | NEXPRCOO | 8 | CHARACTER | &00 NCL PROCEDURE. |
| 48 (30) | NEXFILID | 8 | CHARACTER | FILEID LOGICAL NAME. |
| 56 (38) | NEXFILDD | 8 | CHARACTER | DD NAME. |
| 64 (40) | NEXUAUTH | 1 | HEX | REQUESTING USER'S COMMAND AUTHORITY IN RANGE X'00' TO X'FF'.<br>UNDER EASINET, SET TO X 'FF'. |

```
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - *
*   THE FOLLOWING FIELDS ARE ONLY APPLICABLE TO &FILEID REQUESTS              *
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - *
```

| | | | | |
|---|---|---|---|---|
| 65 (41) | NEXPTYPE | 1 | HEX | FILE PROCESSING TYPE. |
| | NEXPTYP0 | | EQU X'00' | TYPE - STANDARD UDB. |
| | NEXPTYP4 | | EQU X'04' | TYPE - EXT-CHAR NON-UDB. |
| | NEXPTYP8 | | EQU X'08' | TYPE - EXT-HEX NON-UDB. |
| | NEXPTYPC | | EQU X'0C' | TYPE - EXT-ASIS NON-UDB. |
| 66 (42) | NEXUDLEN | 2 | HEX | LENGTH OF DATA IN NEXUDATA. |
| 68 (44) | NEXUDATA | 50 | CHARACTER | DATA PASSED ON &FILE OPEN STMT.<br>AREA WILL BE BLANK PADDED. |
| 118 (76) | NEXFLG1 | 1 | HEX | FLAG BYTE. |
| | NEXF1DYN | | EQU X'01' | FILE WAS DYNAMICALLY ALLOCATED<br>BY USE OF ALLOCATE COMMAND. |
| | NEXF1SYO | | EQU X'02' | REQUESTED FILE IS A SYSOUT FILE.<br>IF ON, NEXUSCLS WILL CONTAIN<br>SYSOUT CLASS BEING REQUESTED. |
| | NEXF1RDR | | EQU X'04' | THIS IS A SYSOUT FILE THAT IS ATTEMPTING<br>TO USE THE INTERNAL READER.<br>NEXF1SYO IS ALSO ON. |
| 119 (77) | NEXSYCLS | 1 | CHAR | IF NEXFLG1/NEXF1SYO IS ON, THIS IS THE<br>SYSOUT CLASS THAT IS BEING REQUESTED. |

```
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - *
*   EQUATES FOR REGISTER 15 RETURN CODES.                                     *
* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - *
```

| | | | |
|---|---|---|---|
| NEXR15RD | | EQU X'00' | READ ONLY. |
| NEXR15UP | | EQU X'04' | UPDATE (INC ADD) WITHOUT DELETE. |
| NEXR15DL | | EQU X'08' | UPDATE INCLUDING DELETE. |

| Offsets | Name | Length | Format | Description |
|---|---|---|---|---|
| NEXR15AB | | | EQU X‘0C’ | ACCESS. |
| NEXR15UA | | | EQU X‘10’ | FILEID NOT AVAILABLE. |
| NEXR15OK | | | EQU X‘00’ | &SMFWRITE REQUEST AUTHORIZED. |
| NEXR15NO | | | EQU X‘04’ | &SMFWRITE REQUEST DENIED. |
| | * THE FOLLOWING FIELD IS ONLY APPLICABLE TO &SMFWRITE REQUESTS. * | | | |
| 65 (41) | NEXSMFRT | 1 | HEX | SMF RECORD TYPE THAT IS THE SUBJECT OF &SMFWRITE REQUEST |
| | * THE FOLLOWING FIELD IS ONLY APPLICABLE TO &EDB, &SQL REQUESTS. * | | | |
| 68 (44) | NEXDBEX1 | 4 | HEX | ADDRESS OF PARAMETER LIST MAPPED BY NEXDB DSECT |
| | * THE FOLLOWING FIELD IS ONLY APPLICABLE TO &NDBOPEN REQUESTS. * | | | |
| 68 (44) | NEXNDEX1 | 4 | HEX | ADDRESS OF PARAMETER LIST MAPPED BY NEXND DSECT |

# L

# MAI Installation Exit MAIEX02

This appendix provides information on how to customize the MAIEX02 exit for use with MAI-OC.

## About the MAI Installation Exit

The installation written exit MAIEX02 is supported by Management Services. 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

## Implementing the MAI Exit

The exit must be link edited and placed into a load library (OS/VS, VM/SP) accessible to Management Services. In OS/390, z/OS, and VM/SP systems the exit must be link edited with the RENT option.

The name of the link edited exit module is identified to Management Services 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 Management Services task.

**Note**

Although executed under a subtask, the termination call to the exit at system closedown is serialized and the termination call must complete before Management Services resumes its termination processing.

### Reentrant Code

You must write the exit to conform with 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 via 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 in an OS/VS system. 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 of Management Services, 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, as the path through MAIEX02 will almost always be 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.

## The Sample Exit

A sample exit, MAIEX02, is distributed in the BASE.INSTALL library. It is recommended that you assemble and study this prior to attempting to write an exit. The macro $NMMAEX2 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.

## Starting 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, a System Initialization call is made. Subsequent MAI session requests will then call the exit. If the exit is not included in RMINIT, but is invoked via command from a Management Services terminal, any existing users with MAI sessions may have to log off Management Services before any calls will be 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 will be zeroed. In addition, MAI will ensure that if any MAI sessions are currently in progress, the ending of those sessions will 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 macro supplied with Management Services, called $NMMAEX2. This macro provides a DSECT expansion to perform the mapping and detailed information on the content of each field.

Standard linkage conventions apply. On entry, the exit must save the contents of all registers (Register 13 will contain 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 instance, they can contain counts or storage addresses. The exit can supply or change the content of any correlator supplied during any call. The correlator names and the ways in which they are used are described below.

### 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 will be returned in all subsequent calls.

### User Correlator

One User Correlator exists for each logon session. That is, when a user first logs on to Management Services, 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 will be 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 will be returned in subsequent calls for that session.

### Security Exit Correlator

In addition, 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. See Chapter 3, *Implementing Security* for further details.

### 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. See Chapter 3, *Implementing Security*, for further details.

## Contents of the Communications Area

The exit has 5 different types of call and the content of the communications area depends on the type of call. The 5 call types are:

- System initialization
- System closedown
- MAI session start
- ACB open
- MAI session end

### System Initialization Call

The System 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.

### System Closedown Call

The System Closedown call is made when Management Services 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 Management Services is about to start a new MAI session. 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 itself

## 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 cancelled
- System Correlator
- User Correlator
- User Security Token
- Session Correlator

- A work area to be used for any purpose
- The count of inbound bytes over the duration of the session
- The count of outbound bytes over the duration of the session
- The count of inbound RUs over the duration of the session
- The count of 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 System Initialization call and the Session Start call, as follows:

### System 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 will be allowed. Any attempts to start an MAI session will be 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 is to 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 will be displayed:

```
SESSION REFUSED BY INSTALLATION EXIT
```

# M

# Delivery of Unsolicited VTAM Messages

This appendix summarizes how Management Services delivers unsolicited VTAM (PPO) messages to OCS terminals. It outlines the decisions made and actions taken when delivering a message.

## Severity Classification of VTAM Messages

VTAM messages are assigned severity classifications. These classifications are as follows (in order of decreasing significance):

- Unsuppressible
- Severe
- Normal
- Warning
- Informational

The classification which pertains to any particular message can be determined by referring to the appropriate VTAM messages manual.

A user can be defined to receive messages of one or more classifications. If a user is defined to receive only one classification of message, messages at or above that classification are applicable. For example, if a user is defined to receive Normal messages, they also receive Severe and Unsuppressible messages.

## Delivery Logic of VTAM Messages

Each VTAM message is delivered according to the following delivery logic:

1. Does the message have delivery options specified in the DEFMSG table, or are unsolicited message delivery options set? If not, ignore message.
2. Does the DEFMSG table specify PPOPROC delivery for this message? If so, queue the message to PPOPROC if active.
3. Does the DEFMSG table specify LOCAL delivery for the message, or have delivery options been changed by PPOPROC to include LOCAL delivery? If not, go to step 10.
4. Is this a message for a particular Network resource? If not, it is classed as Undeliverable. Go to step 7.
5. Deliver the message to all users who have been message partitioned to receive messages concerning the resource named in the VTAM message, and who wish to receive messages of this severity class.
6. Has the message been delivered to any users in point 5? If not, it is classed as Undeliverable.
7. If the message is classed as undeliverable, send it to all users whose profile indicates they should receive undeliverable messages, and who wish to receive messages of this severity class.
8. Deliver the message to all users whose profile indicates they should receive all VTAM messages, and who wish to receive messages of this severity class, regardless of whether or not other users have received it.
9. If the message was not delivered to any users, send it to the system console (that is, *pppp*OPER).
10. If the DEFMSG table indicates REMOTE delivery for this message, or PPOPROC has changed delivery to include remote delivery, deliver to all ISR links which have unsolicited outbound PPO enabled.

**Note**

Undeliverable messages are prefixed by the character U.

A single message can be delivered to any number of users, according to their profile. This allows a systems programmer, for instance, to log on to Management Services and receive all VTAM messages without affecting which messages existing users receive.

If you require all or some PPO messages to be written to the Management Services log, you can profile one of the background users (for example, BSYS) to receive these messages.

Scoped logging of PPO messages can be achieved by profiling a background user ID (for example, BSYS) for the desired messages using NPF. If you want all unsolicited messages logged, use the SYSPARM PPOUSMSG=LOG option.

LOGPROC and a background user MSGPROC can also be used to control message logging.

See the *Network Control Language User's Guide* guide for information on how to use the PPOPROC NCL procedure to alter the message delivery options.

# N

# Performance and Tuning Commands

This appendix provides a detailed description of the performance and tuning commands available to Management Services.

## Using Performance and Tuning Commands

The following series of commands is available to set the Management Services performance parameters and to display the relative activity within a system, of the different categories of process that the system is executing. The default settings are designed to cater for most Management Services installation requirements, but these commands are available to change the defaults if necessary.

The commands should be used with caution and for this reason, they are not documented in the *Management Services Command Reference*.

# SHOW SYSPGT

## Function:

This command displays the summary information about the NCL system performance group tables.

```
SHOW SYSPGT
(0)
```

## Use:

The system performance group table contains historical statistics about each of the four performance groups. It 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.

## Examples:

```
SHOW SYSPGT
```

The information returned is as follows:

**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 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.

## Notes:

The four performance groups are as follows:

- 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 FTS commands.

## See Also:

SYSPGT, SYSRCT, and SHOW SYSRCT commands.

# SHOW SYSRCT

## Function:

This command displays summary or detail information about NCL performance groups.

| | |
|---|---|
| SHOW SYSRCT<br>(0) | [ =SUMMARY \| =*n* ] |

## Use:

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

## Operands:

**=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.

## Examples:

```
SHOW SYSRCT=4
```

The summary information returned is as follows:

**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.

The detailed information returned includes the above statistics for both long term and short term performance control measures, as well as the performance control settings for priorities and forcing procedures to wait, as follows:

**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*.

## Notes:

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 within 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 to 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=.

## See Also:

Performance control commands SYSPGT and SYSRCT, and the display command SHOW SYSPGT.

# SHOW SYSWAIT

## Function:

This command displays Management Services maintask apparent wait statistics.

| |
|---|
| SHOW SYSWAIT<br>(0) |

## Use:

The system maintains wait statistics each time an operating system WAIT is issued. The information returned by this command includes the percentage of elapsed time that Management Services is in an operating system wait (AWAIT%) and the 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.

## Examples:

```
SHOW SYSWAIT
```

The information returned is as follows:

**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.

## Notes:

SYSWAIT statistics should be used as a guide only, as they reflect the voluntary loss of control performed by Management Services and do not reflect operating system resource management activity.

The statistics are maintained on an 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, providing a means for displaying activity in the short-term, compared against the long-term average.

AWAIT% reflects the apparent wait time that Management Services has voluntarily given up control to the operating system. Thus, the actual wait time could be higher, because of such things as page faults or higher priority work.

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 means that work could have been done, but all ready NCL processes were being force-waited by performance controls.

## See Also:

Performance control command SYSRCT and display command SHOW SYSRCT.

# SYSPGT

## Function:

This command is used to set the initial priority for an NCL performance group.

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

## Use:

This command is used as part of performance control. It sets the priority that is assigned initially to procedures running in a performance group.

## Operands:

**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
```

## Notes:

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.

## See Also:

Performance control command SYSRCT and display commands SHOW SYSPGT and SHOW SYSRCT.

# SYSRCT

## Function:

This command is used to set performance control parameters in the system Resource Control Table (RCT).

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

## Use:

This command is used for performance control. It can be used to set *processing unit* trigger values and performance control parameters. Two types of performance control measure are supported:

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

Performance is calculated in two 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.

## Operands:

**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=( 1[ *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.

## Notes:

*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.

Management Services maintains 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 via 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.

## See Also:

Performance control command SYSPGT and display commands SHOW SYSRCT and SHOW SYSPGT.

# O

# Using Management Services with XNF

This appendix describes how to use Management Services with Hitachi's XNF product. It describes the special considerations that apply when Management Services uses XNF, rather than VTAM, as its access method.

The information supplied in this appendix supplements the information provided by this manual. The following sections describe the aspects of Management Services which apply only to XNF, and identify those which do not apply to XNF.

## About XNF

Hitachi's XNF product provides an access method to control an OSI network. In an XNF network, Management Services functionality closely parallels its functionality in a network controlled by VTAM.

When your Unicenter, NetSpy, and SOLVE products run in an XNF network, XNF is used to connect terminals to Management Services, and to connect one system to others located elsewhere in the network. Both types of connectivity use HNA Passthrough protocol, which maintains a degree of compatibility with VTAM-based systems. Terminal connections use HNA LU Type 2 protocol, and INMC connections use the same LU Type 0 protocol as is used in VTAM networks.

### VTAM Versus XNF Terminology

In a VTAM-based system, connectivity between Management Services and VTAM is established by opening an ACB. The ACB is known as the Primary ACB. The name of this ACB is set using the PRI JCL parameter, and can be changed using a SYSPARMS PRIACBNM command.

XNF works in a manner very similar to this. In an XNF-based system, connectivity is established by the XNF ENTRY process, which catalogs a UCE to XNF. The name of the UCE is set by the PRI JCL parameter, and can be modified using SYSPARMS XNFUCENM.

Throughout this manual there are many references to the Management Services *primary ACBNAME* and to *opening the primary ACB*. With an XNF system, you can assume that such references apply also to *XNF UCEname* and to *cataloging the UCE*.

Similarly, any reference to a *VTAM LUname* can be assumed to apply to an *XNF ULEname*, and any reference to a *VTAM APPL* applies to an *XNF ULE* (Type UCE).

Some facilities of Management Services can only be used with VTAM and do not apply to XNF systems. In particular, the following chapters in this manual contain information that does not apply:

- Chapter 8, *Partitioning VTAM Networks*
- Appendix M, *Delivery of Unsolicited VTAM Messages*
- Appendix J, *INMC Security Exit Support*

## Connecting Terminals to Management Services

Any IBM 3270 or compatible terminal, or any remotely-attached 560/20 terminal can be connected to Management Services via the XNF network. Management Services Version 3.1 does not support connection of local 560/20 devices, or devices connected via HSC3.

Use the DEFTERM command to define the attributes of terminals which can connect to Management Services. DEFTERM commands can be included in the RMREADY procedure so they are executed during initialization.

The DEFTERM command defines terminal types as 560/20 or 3270. It also specifies whether or not the terminal supports color and highlighting options and display of DBCS (Kanji) characters.

When connecting a terminal to Management Services, the connection can be initiated from the following:

- A terminal by using XNF's CONNECT command
- Management Services by using the CONNECT command

**Note**

EASINET is not available on XNF systems.

### Specifying Screen Sizes for 3270 Terminals

By default, Management Services supports 3270 terminals as model 2 devices. This means the default screen size is 24 rows by 80 columns. This default size applies to terminal connections initiated both from the terminal and from Management Services. To change the default screen size, use the SYSPARMS XNFSCRSZ operand. If it is changed, the new value is supplied as the alternate screen size in the terminal bind, and is used if possible.

To specify a screen size for individual connections initiated from Management Services, use the SIZE operand on the CONNECT command.

### Extended Attribute Support

If your terminal supports the use of READ PARTITION QUERY, Management Services can obtain extra information about your terminal, such as:

- Color
- Extended highlighting options
- Whether DBCS (Kanji) data is recognized

These attributes override those entered on the DEFTERM command.

To manually change the attributes being used by Management Services for your terminals, use the TERMINAL command.

## Shutting Down and Restarting XNF

If you need to shutdown XNF for maintenance, you also need to disconnect Management Services from XNF, by using the XNF STOP command. This command disconnects all terminals, forces all INMC and APPC connections inactive, drops all NETM connections, and releases the UCE.

When XNF STOP is executed, the following message appears on the console:

```
NC8101 CONNECTION TO XNF RELEASED. REPLY RESTART OR END.
```

All Management Services functions other than communications (including NCL procedures, automation activity, and so on) continue running.

To reconnect XNF to Management Services, reply the RESTART command to the NC801 message. Terminals are able to reconnect and INMC links recover automatically.

## Customizing Management Services

When customizing Management Services in an XNF system, there are JCL parameters and SYSPARMS commands that do not apply. The following sections provide this information.

**Note**

Additional JCL and SYSPARMS commands can be specified in the initialization procedures for Management Services. In XNF the supplied procedures are XNFINIT and XNFREADY.

### JCL Parameters

The PPO and CNM JCL parameters only apply to VTAM-based systems and are ignored if supplied on an XNF-based system.

The PRI JCL is used to set the name of the ULE (Type UCE) to be used for communication with XNF.

## SYSPARMS

The following SYSPARMS operands apply only to VTAM-based systems, and are unavailable on XNF-based systems:

- ACBRETRY
- APPLSTAT
- APPLSTIV
- APPTXT1
- APPTXT2
- APPTXT3
- APPTXT4
- APPTXT5
- APPTXT6
- EASINET
- PPOACBNM
- PPOCOLOR
- PPOHLITE
- PPOPROC
- PPOSOCMD
- PPOSOMSG
- PPOUSMSG
- PRIACBNM
- TPEND
- VDISPLAY
- VTAMAUTH
- VTAMID
- VTAMCDRM
- VTAMNTFY

## Implementing Security in an XNF system

When defining user IDs on an XNF system, the following fields on the UAMS panels should be noted:

- The Terminals to which User is Restricted field on the User Authorities Panel requires you to specify the ULE names of the required terminals.
- The PPO and NPF fields on the OCS Details Panel only apply to VTAM-based systems and if specified, are ignored. These fields include the following:
  - Receive PPO messages
  - Minimum Severity Class
  - NPF Resource List Member
  - NPF Command Member
  - NPF Message Number
  - Undeliverable PPO Messages

## Using INMC Links in an XNF System

INMC is supported on Management Services systems in an XNF network. However, the following points should be noted:

- When defining a static INMC link using the LINK START command, the APPL parameter is used to nominate the name of the UCE to connect to. For example:

  ```
  LINK START=OSAKA APPL=UCE002
  ```

  where `UCE002` is the name of the ULE (Type UCE) defined to XNF and used by the system in Osaka.
- For INMC to function correctly, the ULE (type UCE) entries in the XNF network definition must specify U-FUNCTION-PROFILEs.
- Appropriate ROUTE definitions must exist in the XNF network definition if systems on different hosts are to communicate.
- A standard bind image is used for INMC connections in an XNF network. The bind sets a maximum RUsize of 1024.
- When using ISR, only the AOM conversations class is supported
- XNF systems can use INMC to link to a Management Services system running under VTAM, providing that XCOMS exists to supply the basic connectivity between the VTAM and XNF networks.

**Note**

INMC/EF is not supported in an XNF network.

## Using APPC Links in an XNF System

APPC can be used to communicate through XNF with another system which is also using XNF for communications. The following points should be noted:

- Management Services supports single session links and parallel session links through XNF.
- When defining an APPC link using the LINK START command, the LU operand is used to nominate the name of the XNF ULE (type UCE) to connect to.
- If a Mode Control Table (MCT) is defined, the Logmode name is not used.

## Issuing Management Services (MS) Commands on an XNF System

The following commands are supported only when the XNF access method is used:

- CONNECT
- XNF STOP
- XNFTRACE
- SHOW XNFTRACE

These commands are described in the *Management Services Command Reference*.

Some commands that are available when Management Services is using the VTAM access method are *not* available with XNF. These are:

- ACT
- ACTLOGON
- D
- DEFLOGON
- DEFMSG
- DELLOGON
- F
- FORCE
- INACT
- NPTAB
- OPNDST
- PPO
- R
- REPLOGON
- REPLY
- SHOW DEFMSG

- SHOW DEFLOGON
- SHOW PPOSTATS
- SHOW NPTAB
- SHOW NP
- SPO
- SUSLOGON
- TRACE/NOTRACE
- V

## Using NCL Components in an XNF System

The following NCL components relate specifically to systems using the XNF access method:

- &ZUCENAME—a system variable used to determine the UCE name which Management Services is using to communicate with XNF
- &ZAMTYPE—a system variable used to determine the name of the access method used to connect the terminal on which the NCL procedure is running
- &ZAMCHECK—a built-in function used to determine whether support is enabled in Management Services for a particular access method

See the *Network Control Language Reference* for more information.

# P

# Defining User Migration ID Exits

This appendix describes user migration ID exits for Extended MCS (EXTMCS) consoles.

## 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 exit.

For information about using these exits to customize your Management Services region, see Chapter 2, *Customizing Management Services*.

### 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 with SYSPARMS AOMSUBCH). 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 OS/390 commands.
- Forcing YES or NO in all cases (these are fairly trivial exits: a simple LA R15,4 (or 8) or a BSM 0,14).

### Sample Exit

A sample exit, NMMIGIDX, is distributed in source form. It illustrates the coding required to produce a useful exit.

### Activating the User Exit

To activate the user exit, specify an exit name in the CONSOLES parameter group in ICS.

### User Exit Requirements

The migration ID determination user exit must meet the following requirements:

- The exit must be assembled and link-edited into a Management Services-accessible load library. Any name is suitable (the name is specified in the SYSPARMS command to activate the user exit).
- Because Management Services executes APF-authorized, 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 Management Services processing.
- If the exit abends, it is disabled.

## Input Parameters for User Exits

The migration ID exit is called with a parameter list. The format of this list is shown below.

**Note**

Although the exit is called in whatever AMODE it is linked with, all parameters are placed below 16M.

R1 points to a list of fullword addresses. These addresses in turn point to parameters.

```
R1  ==> A(PARM1)
        A(PARM2)
        A(PARM3)
        A(PARM4)
        A(PARM5)
        A(PARM6+X'80000000')   (end of list)
```

The parameters are:

**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 nonblank character. This is the complete command (uppercased).

**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 nonblank 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 was detected, the second (actual) command buffer address is the address of the target command within 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.

## Exit Tracing

You can trace migration ID exit processing (for either the default exit or a user exit) by using SYSPARMS AOMCOPTS=01. This causes a message (N86M01) to be written to the log for each request for analysis of a command (either by using SYSCMD MIGID=EXIT or by using the NCL built-in function).

**Note**

The exit is not called if no consoles are acquired or if the current console type does not support migration IDs (only EXTMCS consoles support migration IDs).

# Glossary

This glossary defines the terms and abbreviations commonly used with Management Services.

It also includes references to terms used in an IBM environment and any equivalent Fujitsu terms.

**3270 VDU terminal**
An IBM video display terminal. This is often used to refer to the entire range of 3270 terminals. When followed by a number (for example, 3270-5), a specific model is intended.

**370/390**
This is an abbreviation for IBM's System 370 or S/370 architecture. It is often used to indicate any mainframe CPU that implements this architecture.

**3705/3725/3745/3746**
An IBM front end communications processor (the Fujitsu equivalent is a CCP or 2806).

**9526**
A Fujitsu video display terminal.

**ACB (Access Method Control Block)**
A control block that links an application program to an access method such as IBM's VTAM or VSAM.

**ACB-sharing**
MAI's ability to use a single VTAM ACB for multiple sessions.

**Access Security Exit**

An installation-provided routine that may be used to replace the Management Services UAMS functions, partially or completely, allowing logon, logoff, and password maintenance requests to be passed to an external security system.

**ACF/VTAM (Advanced Communication Facility/VTAM)**

IBM's product implementation of SNA's SSCP or CP.

**Activity Log**

A system-maintained log that records all important activity for use in later problem determination.

**Alternate Index**

An alternative view of the data contained within a VSAM keyed dataset. The alternate index allows data to be retrieved using an alternate key in addition to the usual access through the primary key.

**AOM (Advanced Operation Management)**

A facility of Management Services that manages and controls local and remote operating systems.

**AOMPROC**

The name given to an NCL procedure used to intercept messages from the screening table component of AOM to provide extended message processing.

**APF (Authorized Program Facility)**

Describes the special authorization level required within the operating system for certain applications.

**APPC (Advanced Program to Program Communications)**

An IBM-defined application level protocol which makes use of SNA's LU 6.2. It is an accepted industry standard for transaction processing between peer systems.

**APPL (Application Program)**

A VTAM term used to describe the definition that allows an application to use VTAM facilities.

**APPN (Advance Peer-to-Peer Networking)**

IBM's data communications support that routes data in a network between two or more APPC systems that do not need to be adjacent.

**Application Plan**

A DB2 term that refers to the control structure used by the database to allocate resources and execute SQL statements.

**ASN.1**

Abstract Syntax Notation One, defined by ISO 8824, is an abstract syntax used to describe data structures. It is used by Mapping Services to define data structures within Management Services.

**Authorization ID**

A DB2 term that refers to the ID used by DB2 to control access to a database.

**BCI (Batch Command Interface)**

A subcomponent of EIP that allows commands to be issued from batch jobs into a Management Services system.

**BER (Basic Encoding Rules)**

The transfer syntax used by Mapping Services to serialize data for transmission. It is defined by ISO 8825.

**BIND**

1. A VTAM term describing the action of logically linking one network resource with another network resource.
2. A DB2 term for the process that connects together the application plan and the Database Request Module (DBRM) used by an application.

**Broadcast Services**

The message broadcasting function provided by Management Services.

**Build/Verify**

A semiautomatic method of adding or verifying a series or group of resource records, by retrieving data from the hardware and software in the live configuration.

**CAF (Call Attach Facility)**

A connection technique provided by IBM and used by the EDBS facility of Management Services, to communicate with DB2.

**CAS (Common Application Services)**

CAS functions are a collection of NCL routines designed to facilitate program development.

**CICS (Customer Information Control System)**

An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs.

**Client**

A functional unit that receives shared services from a server.

**CNM (Communications Network Management)**

IBM term for its SNA management facilities.

**CNMPROC**

The name given to an NCL procedure used to intercept CNM records received across the VTAM CNM interface by the NEWS component of NetMaster for SNA.

**Command Partition**

A term associated with network partitioning that describes the group of network resources a user ID is authorized to reference with VTAM commands.

**Control Member**

A term associated with network partitioning that describes the list of resource table names applying to a user ID. This control member is referenced in the definition of USERID.

**CP (Control Point)**

A collection of tasks which provide directory and route selection functions for APPN.

**Cross-domain Resource (CDRSC)**

A VTAM term describing the definition of a network resource that is owned by VTAM in another domain.

**DBCS (Double Byte Character String)**

Refers to a mode of representation of data where each byte of data requires 16 bits rather than 8 bits as required by Single Byte Character String. DBCS is used for the implementation of Asian languages such as Japanese.

**Deferred Write**

A performance option for use with UDBs to minimize I/O by deferring the writing of records.

**DEFLOGON**

The term used to describe an application entry path to be supported by SOLVE:Access. The DEFLOGON command is used to define application access paths and their associated text strings.

**Dependent LU**

Any logical unit that is made active by a command from the host system over a data link.

**Dependent Processing Environment**

An NCL processing environment which is a child process to another NCL process and hence has its output delivered as input to some other NCL procedure.

**DOM (Delete Operator Message)**

A non-roll deletable message (NRD) can be deleted from a window only when a DOM is issued.

**Domain**

1. An SNA term describing a domain that consists of the set of SNA resources controlled by one common control point called an SSCP. In terms of implementation, an SSCP is the host access method (VTAM). An SNA network consists of one or more domains.
2. A VTAM term that describes a logical division of a network. Networks are divided into domains that are associated with the way they are controlled.
3. A term that describes any instance of Management Services that is running.

**Domain ID**

A 1 to 4 character mnemonic used as a unique identifier for a domain.

**Dynamic Allocation**

Assignment of datasets to a program at the time the program is executed rather than at the time the job is started.

**EASINET**

A component of SOLVE:Access. EASINET allows idle terminals to be brought under the control of SOLVE:Access and to be operated by installation-written NCL procedures that can provide a wide range of *front end* facilities to end-users of the network.

**EDBS (External Database Support)**

Facility allowing NCL to use DB2 databases.

**EDS (Event Distribution Services)**

A Management Services facility for notifying NCL procedures of events.

**EIP (External Interface Package)**

A facility that allows you to connect to, and issue commands on, a Management Services system from external sources such as TSO and BATCH jobs.

**ER (Explicit Route)**

The physical path between two network nodes. (SNA)

**ESDS (Entry Sequenced Data Set)**

A non-keyed VSAM dataset whose records are stored and retrieved in sequential order, and new records added to the end of the data set.

**Exit**

An installation-written routine that can be driven from a point within a program to provide data to the program, or perform additional processing relevant to that installation's specific requirements.

**Extended Datastream**

A 3270 datastream containing fields that utilize color and/or extended highlighting capabilities of the terminal.

**External Database Connection**

A term used to refer to connections from Management Services to the database which use the Call Attach Facility (CAF) to connect to DB2. NCL connections are connections to DB2 that utilize the Management Services connection. See also *Management Services Connection* and *NCL Connection*.

**External Database ID**

The ID of the external database as it is known to the operating system.

**FTS (File Transmission Services)**

A feature of Management Services which is used to transmit datasets between CPUs.

**Function Key**

A key on a terminal's keyboard which causes a panel to be completed. In the case of 3270, this term also applies to the ENTER key (the same behavior can be achieved on a 6530).
Also called Program Function (or PF) keys.

**ID**

Identification

**IMS (Information Management System)**

IBM's database/data communication (DB/DC) system that provides a hierarchical database manager and transaction processing system.

**Independent LU**

A logical unit that does not receive an active LU over a link. Can act as a primary LU or secondary LU, and can have one or more LU-LU session at a time.

**Initiator**

The component of FTS which schedules transmission of a dataset.

**INMC (Inter-Management Services Connection)**

This facility allows systems running in a network to communicate with each other, providing general-purpose data transfer within the network. INMC provides the capability for up to sixteen sessions between any pair of systems. In appropriate systems, these sessions can traverse different physical network paths, thus increasing throughput. INMC also provides link security and management facilities.

**Internet**

A wide area network connecting networks all over the world. Uses TCP/IP as the standard for information transmission.

**I/O**

Input/Output.

**IS**

Information System.

**ISR (Inter-System Routing)**

Provides centralized control at the system level through INMC.

**KSDS (Key Sequenced Data Set)**

A VSAM dataset whose records are directly accessed by a user-supplied key.

**LAN (Local Area Network)**

A computer network within a limited geographical area. Is not subject to external regulations.

**Link**

A term used to describe a logical connection between two peer communications systems such as two Management Services systems. See also *INMC (Inter-Management Services Connection)*.

**LOGMODE**

A VTAM term used to describe a set of characteristics and protocols of a logical unit.

**Logon-path**

A path through which users of SOLVE:Access gain access to other VTAM application programs. Paths are defined and controlled by the DEFLOGON command.

**LOGPROC**

The name given to an NCL procedure used to process messages destined for the Management Services activity log.

**LSR (Local Shared Resources)**

A technique for buffering I/O to VSAM files called LSR pools. NCL supports this type of processing for User Databases (UDBs).

**LU (Logical Unit)**

SNA introduced the concept of the logical unit (LU). The LU is a type of SNA network-addressable unit (NAU) that provides protocols for end users to gain access to the network and to the functional components of the LUs.

**LU0**

An unconstrained SNA protocol that allows implementers to select any set of available protocol rules, as long as the two LUs are able to communicate with each other successfully according to the rules chosen. Therefore, all LU types are an implementation of LU Type 0.

**LU1**

A line-by-line or typewriter type terminal (for example 3767, 3770), using SNA protocols.

**LU2**

A 3270 type terminal using SNA protocols.

**LU3**

LU Type 3 was implemented to support printers with a different data stream format. LU Type 3 is used by printers attached to an IBM display cluster controller.

**LU4**

LU Type 4 was implemented so that office system products could transfer documents.

**LU6.2**

A protocol that serves as a port into an SNA network. LU6.2 defines a specific set of services, protocols, and formats for communication between logical processors. LU6.2 provides presentation services for presentation of data to the end user, transaction services for performing transaction processing on behalf of the end user and LU services for managing the resources of the LU.

**LU7**

An SNA protocol that is used by word-processing devices.

**MAI (Multiple Application Interface)**

A facility of SOLVE:Access which is used to provide sessions with any number of other VTAM application programs from one terminal.

**MAI/EF (MAI/Extended Function)**

A SOLVE:Access facility which provides an extension to NCL to support session scripts which allow an NCL procedure to control the application and terminal session flow. MAI/EF also includes the Session Replay Facility (SRF) which allows recording and replay of session scenarios, and Screen Image Services (SIS) which allows recording of screen images for later retrieval or to send to another user.

**MAI-FS (MAI Full Screen)**

A SOLVE:Access facility that allows a single terminal to be used to provide full screen access to any number of other applications. The user may 'jump' from one application to another using designated 'jump' keys on the keyboard, or special command strings.

**MAI-OC (MAI Operator Control)**

A Management Services facility that allows an operator in OCS to have LU Type-1 sessions with many other applications from an OCS window. When used in conjunction with MSGPROC NCL procedures this can provide automated central monitoring and operation of multiple applications from the one operator console.

**Management Services**

This the central core of functions and service routines that supports all of the NetMaster, NetSpy, and SOLVE products.

**Management Services Connection**

A Management Services connection is a communication link from Management Services to the external database. It is started, stopped, defined, and deleted using the EDB command. A Management Services connection must be defined and started before opening an NCL connection. See also *NCL Connection*.

**Mapping Services**

A facility of Management Services that enables programmers to define complex data structures for use by applications.

**MDS-MU (Multiple Domain Support-Message Unit)**

The message unit used in data transmission between management applications in SNA networks.

**MDO (Mapped Data Object)**

Any data item that can be represented as a continuous string of bytes in storage.

**Message Partition**

A term associated with network partitioning that describes the group of network resources for which a user ID will receive unsolicited (PPO) VTAM messages.

**MIB (Management Information Database)**

Any database that provides information about the structure and capabilities of a management application.

**Modify Interface**

A means of communicating with an application program from the system console in OS/VS systems. The MODIFY command, abbreviated to F, avoids having an outstanding REPLY at the system console. Management Services supports the use of the MODIFY command.

**MSGPROC**

An NCL procedure used to intercept and process messages destined for a user's Operator Console Services (OCS) window.

**MSP**

An operating system for large scale Fujitsu systems.

**NAU (Network Addressable Unit)**

In SNA, a logical unit, a physical unit, or a system services control point. The NAU is the origin or destination of information transmitted by the path control network.

**NCL (Network Control Language)**

The interpretive language that allows logical procedures (programs) to be developed externally to Management Services and then executed by Management Services on command. NCL contains a wide range of logic, built-in functions and arithmetic facilities which can be used to provide powerful monitoring and automatic control functions.

**NCL Connection**

An NCL connection is a communication link from an NCL procedure to the external database. An NCL connection is opened and closed with the &EDB verb. A Management Services connection must be defined and started before opening an NCL connection. See also *Management Services Connection*.

**NCL Procedure**

A member of the procedures dataset comprising NCL statements and Management Services commands. The NCL statements and other commands are executed from an EXEC or START command specifying the name of the procedure.

**NCL Process**

The NCL task that is invoked, usually by a START command to execute one or more associated procedures. Each NCL process has a unique NCL process identifier.

**NCL Processing Environment**

Provides the internal services and facilities required to execute NCL processes for the user, from its associated Management Services window.

**NCL Processing Region**

All users (real or virtual) have an NCL Processing Region associated with their user ID while logged on. This region provides all of the internal services needed to allow the user to have processes executed on their behalf.

**NCLID**

A 6-digit NCL process identifier which is unique within the system. It is used to identify a process for the purpose of communicating with that process.

**NCP (Network Control Program)**

This resides within and controls the operation of a communications controller. The NCP communicates with VTAM.

**NCPView**

A component of NetMaster for SNA that allows monitoring of NCP configuration.

**NCS (Network Control Services)**

A facility of NetMaster for SNA that allows display and control of SNA network resources.

**NDB**

NDBs are field oriented databases in which data is stored and retrieved in named fields. They are designed for environments that have complex, high volume data storage and retrieval requirements.

**NEWS (Network Error Warning System)**

A facility of NetMaster for SNA which is used to provide network error and traffic statistics and error alert messages.

**NMVT (Network Management Vector Transport)**

A request/response unit (RU) that flows over an active session between a physical unit (PU) and a control point (CP).

**Node**

A connection point in a communications network.

**NPF (Network Partitioning Facility)**

A facility of Management Services that allows the range of resources which an operator can influence to be denied.

**NPF Control Member**

A member of the NPF dataset which defines a list of member names that are to be the resource tables for the associated user ID.

**NPF Resource Table**

A member of the NPF dataset that defines a group of network resource names. The resource names can be defined specifically or generically using wildcard characters. A resource table is addressed via a control member.

**NRD (Non-Roll Delete) Message**

A message that will not roll off an OCS window display until explicitly deleted. See *DOM*.

**NT 2.1**

Node Type 2.1. A node in an SNA network. It implements a peer-to-peer protocol and allows greater dynamics in network configuration, greater independence in session set up between partner LUs and reduced definitions. Same as PU type 2.1.

**NTS (Network Tracking System)**

A facility of NetMaster for SNA used to provide SNA session monitoring, dynamic online network tracing, accounting, and response time information in conjunction with diagrammatic representations of session partners.

**Object Services**

An object-oriented development environment used to define and maintain the application data and methods of some of the SOLVE products.

**OCS (Operator Console Services)**
A facility of Management Services that provides general operational control and an advanced operator interface to VTAM for network management.

**OS/390**
An IBM operating system.

**OSI (Open Systems Interconnection)**
A set of ISO standards for communication between computer systems.

**Packet**
The unit of data used in transmission.

**Panel Maintenance**
A facility of Management Services which allows users to generate and modify panel definitions used for presentation purposes by NCL procedures. In releases prior to Version 3.0, this function was known as Edit Services.

**Panel Services**
A facility of Management Services for displaying full-screen panel definitions.

**Panel Skip**
The ability to chain menu selection requests together without having to display intermediate selection panels.

**Password**
A 1- to 8-character string chosen by a user and linked to their user ID for security purposes. To gain access to the system a user must enter both their defined user ID and its associated password.

**PDS (Partitioned DataSet)**
A type of dataset format that supports named data segments in the one physical dataset.

**PFK (Program Function Key)**
See *Function Key.*

**PIU (Path Information Unit)**
An SNA packet.

**PLU (Primary Logical Unit)**
Relates to SNA. A type of LU that is usually used by the application programs in a host. It refers to the BIND sender for a session. See also *Primary and Secondary.*

**PPI (Program-to-Program Interface)**
PPI is a general-purpose facility which allows programs, written in any language, to exchange data.

**PPO (Primary Program Operator)**

A VTAM term that describes a facility of VTAM that allows unsolicited network messages to be delivered to an application program, such as Management Services, for processing.

**PPOPROC**

The name given to the NCL procedure used to process unsolicited VTAM (PPO) messages.

**Preload**

A term applied to NCL procedures which are loaded into the system before being required, to improve system performance.

**Primary Menu**

The first menu of an application.

**Primary and Secondary**

Primary and secondary are SNA terms for describing the LU's role when the session is established. The primary LU sends the BIND request that causes the session to be established, and the secondary LU receives the BIND request. Rules defined in the BIND request determine which of these is the first speaker in the exchange of information.

**PSM (Print Services Manager)**

PSM is a facility of Management Services which simplifies the control of the physical printing of reports on JES or network printers.

**PU (Physical Unit)**

The part of a control unit or cluster controller which fulfils the role of an SNA-defined physical unit. Each node (a logical grouping of hardware) in an SNA network is addressed by its PU. There are 4 types of nodes or PU in an SNA network: PU-T5, PU-T4, PU-T2, PU-T1. A PU is a type of NAU. (See *NAU (Network Addressable Unit)*.

**PU Type 1**

A type of Physical Unit or Node in an SNA network. Consists of a terminal (such as an IBM 3278).

**PU Type 2**

A type of Physical Unit or Node in an SNA network. Consists of a cluster controller (such as an IBM3274, 3276, 3770 or 3790).

**PU Type 4**

A type of Physical Unit or Node in an SNA network. Consists of a communications controller (such as an IBM 3704, 3705, 3725 or 3745).

**PU Type 5**

A type of Physical Unit or Node in an SNA network. Consists of a host computer system (such as an IBM30*xx*, S/370 or 43*xx*, running VTAM).

**PVC**

X.25 Permanent Virtual Circuit.

**Report Writer**

A facility of Management Services which allows the creating and customizing of report definitions.

**Request Unit (RU)**

A message unit in an SNA network that contains control information such as a request code, or function management headers, end-user data, or both.

**Reserved word**

The term given to a token that will terminate an input expression if found unquoted outside the topmost parenthesis level. In the context of a particular verb statement the verb keywords are reserved words. A reserved word has special meaning for the current statement only, and different statements have different reserved words.

**Resource Table**

A term associated with NPF that describes a list of resource names or generic resource names that define a command or message partition.

**Response Unit (RU)**

A message unit in an SNA network that acknowledges a request unit.

**Return Code**

A code returned from the system that indicates the success or failure of the task performed.

**RMINIT**

The NCL procedure automatically executed after system initialization has completed. It cannot contain commands that require VTAM facilities as it is executed before the primary ACB is opened. The procedure name can be changed by the installation.

**RMREADY**

The NCL procedure automatically executed once the primary ACB is open. It can contain commands that require VTAM facilities. The procedure name can be changed by the installation.

**ROF (Remote Operator Facility)**

A facility of Management Services that allows an operator to sign on to a remote location, execute commands and have the results returned.

**RSM (Resource Status Monitor)**

A generic facility for monitoring and controlling and resources (not only SNA resources) for which configuration information is available.

**RTM (Response Time Monitor)**
A facility provided by IBM's 3*x*74 control units to monitor end-user response times. NEWS can interpret this data .

**SAW (Session Awareness Data)**
Network management data supplied by VTAM and processed by NTS.

**SDLC (Synchronous Data Link Control)**
A discipline for managing information transfer over a communications link.

**Sequence Number**
A number assigned to each message exchanged between a VTAM application program and a logical unit. Values increase by one throughout the session, unless reset by the application program using an STSN or CLEAR command.

**Server**
A process designed to serve the data to a client, or request process, for one or more users.

**Session Name**
A name assigned to a workstation or session to permit it to receive messages or share resources.

**SIS (Screen Image Services)**
A part of the MAI/EF facility of SOLVE:Access which provides the ability to record screen images for later retrieval or to send to another user.

**SIS (Sequential Insert Strategy)**
A technique that governs the insertion rule for data in VSAM KSDS. Normally used when ascending keys are being added to a dataset, to help in efficient use of space within the dataset.

**SLU (Secondary Logical Unit)**
(SNA) A type of LU that is usually used by the end-users at the terminals or by programs which reside in the peripheral node. See also *Primary and Secondary.*

**SMF (System Management Facility)**
An optional control feature of OS/VS that provides the means for gathering and recording information that can be used to evaluate system usage.

**SMP (System Modification Program)**
An IBM program used to install software on OS/VS1 and OS/VS2 systems.

**SNA (Systems Network Architecture)**

This term describes the logical structure, formats, protocols, and operational sequences for transmitting communication data through the communication system (Fujitsu equivalent is FNA). A set of standards that allows the integration of all the different IBM hardware/software products into a universal network. Introduced in 1974.

**SNI (SNA Network Interconnection)**

The connection of independent SNA networks using gateways.

**SNMP (Simple Network Management Protocol)**

An Internet standard for network management.

**SPO (Secondary Primary Operator)**

A VTAM facility that allows solicited network messages to be delivered to an application program.

**SRF (Session Replay Facility)**

A part of the MAI/EF facility of SOLVE:Access which provides the ability to record and playback terminal session scenarios.

**SSCP (System Services Control Point)**

A part of VTAM and the focal point of SNA networks. It controls general management of each domain.

**Structured field**

Representation of user ID attribute information exchanged between Management Services and its security exit.

**Subtask**

A unit of work that is established by a main task and is displaceable by the operating system.

**SVC**

X.25 Switched Virtual Circuit.

**SYSPARMS**

System parameters—values that affect Management Services system capabilities. Most SYSPARMS can be modified dynamically.

**TCP/IP (Transmission Control Protocol/Internet Protocol)**

A set of communications protocols that support peer-to-peer connectivity functions for both LANs and WANs.

**TSO (Time Sharing Option)**

An IBM product that allows terminal operators to interact directly with computer resources and facilities. Used mainly by application and system programmers. (Fujitsu equivalent is TSS).

**UAMS (Userid Access Maintenance Sub-system)**

The security component of Management Services that supports the definition of authorized users and their associated function and privilege levels.

**UCS (Universal Character Set)**

A printer feature that permits the use of a variety of character sets.

**UDB (User Data Base)**

1. UDB file access method layer allowing file access from NCL.
2. A term used to identify VSAM datasets to which NCL procedures may have access using the &FILE verb (GET, PUT, ADD, and DEL options).

**User ID**

Defines the function and privilege level to which a specific user is entitled when they sign on to the system. It is associated with a secret password to prevent use by unauthorized personnel. This definition is stored in the UAMS dataset or on an external security system.

**User Services**

A facility of Management Services that allows the creation of a range of specific procedures for different users, or classes of user, if required.

**USS (Unformatted System Services)**

A VTAM term that describes a facility that translates an unformatted command such as LOGON or LOGOFF, into a field formatted command for processing by formatted system services. Applies to terminals before connection to an application.

**Verb**

The term given to a stand-alone statement in an NCL program. NCL *verbs* cause actions to occur. There are different types of verbs, some that dictate the flow of processing and logic, others that fetch information for the procedure to process and others that cause data to flow to external targets.

**VFS (Virtual File Services)**

The VSAM dataset, used by many facilities as a database.

**VM (Virtual Machine)**

A superset operating system that allows other operating systems to run as if they each had their own machine.

**VOS3**

A Hitachi operating system comparable to IBM's MVS.

**VR**

(Session) Virtual Route.

**VSAM (Virtual Storage Access Method)**
A method for processing data files that utilizes relative, sequential, and addressed access techniques.

**VSE (Virtual Storage Extended)**
An operating system for small IBM systems.

**VTAM (Virtual Telecommunications Access Method)**
A suite of programs that control communication between terminals and application programs.

**WAN (Wide Area Network)**
A network that provides communication services to a large area, for example, a telephone network.

**Wildcard**
The term used to describe the character used (usually an asterisk) when defining resources generically—no specific matching character is required in the wildcard character position.

**X.25**
An international recommendation regarding the connection of computer equipment to public data networks.

**XNF**
A Hitachi network access method for OSI networks.

**XSP**
A Fujitsu operating system—successor to FSP.

**z/OS**
An IBM operating system.

# Index

## Symbols


$CACNTL 4-12
$CMDENT 16-16
$LOBROW procedure 12-3
$LOPROC procedure 12-3
$NDJPROC procedure D-22
$NMLOCK procedure D-23
$NMNCEX1 macro 4-13
$NMNCPL macro 15-3
$NMPMENU procedure D-27
$PSBA80X banner exit 11-20
$PSDS81X
    printer exit for a dataset 11-28
$PSML78X
    printer exit for NET/MAIL 11-20
$PSML79X
    printer exit for NET/MAIL 11-20
$PSPOOL 4-12
$SANMLSR
    sample LSR pool definition 16-9
$USERSER procedure E-6
&CALL 16-17
    extensions 15-3
    requests 15-2
    sharing subpool zero D-15
    using
        with SUBSYS facility 15-2
        without SUBSYS facility 15-2
&CONTROL RESCAN 16-16
&DOM
    statement 17-7
    verb 17-9
&FILE CLOSE statement 4-12
&FILE OPEN statement 4-11
&FILERC system variable 4-11
&INTCMD verb 6-25, 12-13, E-13
&INTREAD verb 6-25
&LOCK verb 15-2, D-23
&LOGCONT verb 12-3
&PANEL verb 3-22
&RETCODE verb 17-10
&RSCCHECK built-in function 8-6
&SECCALL verb 3-17, E-11, F-18
&SMFWRITE
    authorization exit D-28
    return codes 4-16
    statement 4-11, D-42
    verb D-28
&USERPW system variable D-51
&WRITE verb E-12
&WTO
    statement 17-5
    verb 17-9
&WTOR
    statement 17-6
    verb 17-9
&ZAMCHECK built-in function O-8
&ZAMTYPE system variable O-8

&ZDOMID
system variable 17-7
verb 17-10
&ZFDBK verb 17-10
&ZMAOMAU system variable 17-9
&ZMAOMBC verb 17-9
&ZMAOMDTA verb 17-9
&ZMAOMID verb 17-9
&ZMAOMJI verb 17-9
&ZMAOMJN verb 17-9
&ZMAOMMID verb 17-9
&ZMAOMMIN verb 17-9
&ZMAOMMLC verb 17-9
&ZMAOMMLD verb 17-9
&ZMAOMMLE verb 17-9
&ZMAOMMLL verb 17-10
&ZMAOMMLT verb 17-10
&ZMAOMMLV verb 17-10
&ZMAOMMSG verb 17-10
&ZMAOMRC verb 17-10
&ZMAOMRCM verb 17-10
&ZMAOMRCX verb 17-10
&ZMAOMSOS verb 17-10
&ZMAOMTM verb 17-10
&ZMAOMTYP verb 17-10
&ZMAOMUFM verb 17-10
&ZMPTEXT system variable 17-5
&ZUCENAME system variable O-8
&ZUSERLC system variable E-2

# A

ABENDCD operand D-12
AC (Action) action 2-7
ACBRETRY operand D-12
activity log
browsing 12-5
cross referencing 12-15
dealing with I/O errors 12-16
defining lines per page D-23
flushing 12-11
format 12-12, 12-13
hardcopy log 12-13
in OS/VS or VM/SP systems 12-11
logged information 12-2
online swapping 12-6
recording additional information 12-7, 12-16
specifying procedure D-23
swapping 12-14
ALLOC command 12-15
ALLOCATE command 3-22
AM JCL parameter B-3
AOM START command 17-8
AOM STOP command 17-8
AOMCMIGI operand D-59
AOMCMIGX operand D-59
AOMCOPTS operand D-59
AOMCPREF operand D-60
AOMCTO1 operand D-60
AOMCTO2 operand D-60
AOMCTO3 operand D-60
AOMCTYPE operand D-61
AOMCUTOK operand D-61
AOMGQSZ operand D-62
AOMJESCH operand D-62
AOMMLTO operand D-62
AOMPQSZ operand D-62
AOMPRFJI operand D-63
AOMPRFJN operand D-63
AOMPRFMN operand D-63
AOMPRFSN operand D-63
AOMPRFTM operand D-63
AOMPROC procedure 6-28
AOMSSID operand D-63
AOMSUBCH operand D-64
AOMTRACE operand D-64
AOMTRCRC operand D-64
AOMTRLIM operand D-64
APPC 7-2
control table
defining 7-9
interactions 7-8
requirements 7-7
conversation types 7-4
default modes 7-15
definition examples 7-24
within same system 7-25
Dynamic Link Table (DLT) 7-12
establishing sessions 7-22
implementing security 7-16
from a remote system 7-18
region use 7-19
links
activating automatically 7-20
activating manually 7-19
deactivating manually 7-19

defining 7-6
Management Services to Management Services session 7-25
Management Services to remote node 7-26
log modes 7-14
LU type 6.2
option sets 7-4
sessions 7-2
mapped conversations 7-4
Mode Control Table (MCT) 7-13
modes for session links
parallel sessions 7-15
single session 7-14
nominating mode names 7-13
Option Set Control Table (OSCT) 7-13
parallel session links 7-23
selecting
destinations 7-11
sessions for conversations 7-23
session initiation 7-20
session polarity 7-3
single session links 7-23
tables
Dynamic Link 7-12
Mode Control 7-13
Option Set Control 7-13
TCT, defining 7-9
transactions
generic 7-10, 7-28
qualified 7-10, 7-27
using in XNF systems O-7
application status time interval D-13
APPLSTAT operand D-12
APPLSTIV operand D-13
APPTXT1 operand D-13
APPTXT2 operand D-13
APPTXT3 operand D-13
APPTXT4 operand D-14
APPTXT5 operand D-14
APPTXT6 operand D-14
ARMNAME JCL parameter B-3
ATTACH command 15-2
authority levels, commands 2-13, H-1
summary table H-2
AUTOEXEC operand D-14
automatic log swapping 12-16

# B

background
environments
defining user IDs 3-15
MSGPROC procedure 3-16
using with ROF 3-16, 6-25
processes
defining user IDs 3-15
types 3-15
BASE.INSTALL library A-6
BIND command 7-21
BIND RUSIZE 16-14
Broadcast Services
customizing requirements 10-1
panels
Group Definition 10-3
Group List 10-2
Resource Definition 10-4
Resource List 10-4
system groups
accessing 10-2
adding 10-3
listing 10-2
resource definitions
adding 10-4
listing 10-3
types of resources 10-5
buffer sharing 16-9
built-in functions
&RSCCHECK 8-6
&ZAMCHECK O-8

# C

calls of user ID security exit F-6
CALLSHR0 operand D-15
CDELAY operand D-15
centralized control, Management Services domains 6-24
ROF 6-24
centralized control, MS domains
ISR 6-25
CICS/VS, with MAI-OC 9-5
CISIZE 16-11
clearing printer spool 11-18
CM001 command member 8-4
CMDAUTH operand 2-12, 2-13, D-15, H-2
CMDREPL

operand D-16
CMDREPL, generic operands
command D-16
CMDREPLS parameter group 2-14, 2-15
CMDSTACK operand D-16
CMS FILEMODE 8-4
CMS files 12-12
CNM JCL parameter B-3
CNMACBNM operand D-53
CNMNETV 13-13
CNMPROC
operand D-53
procedure 6-28
CNOS transaction 7-23
command authority, changing D-15
command partitioning 8-2
defining resource tables 8-8
resource group 8-6
specifying in user ID definition 8-6
command stack size D-16
commands
authority levels H-1
customizing 2-13
summary table H-2
for performance and tuning N-1
for PPI 13-15
for SOLVE SSI 13-11
for UTIL0028 utility C-3
LINK START 6-8
SYSCMD
&INTCMD environment 17-4
MVS, OS/390, MSP, VOS3 17-2
VTAM display 8-3
commands, specific
ALLOC 12-15
ALLOCATE 3-22
AOM START 17-8
AOM STOP 17-8
ATTACH 15-2
BIND 7-21
CMDREPL D-16
DD C-4
DEFTERM A-5
DELLINK 7-12
DELOPSET 7-13
EQUATE 2-15
ERROR C-3
EVERY D-19
LINK 6-6
LINK RESET 6-9
LINK START 7-19
LINK STOP 6-9
LIST C-4
LOCK D-23
LOGSWAP 12-12, 12-15
NPTAB 8-3, 8-11
PARMSEP C-3
PF 2-8
PFLIST 2-8
PGM C-3
PPREF C-3
PROFILE 17-8, D-20, E-12, E-13
PSUFF C-3
REPLINK 7-12
REPOPSET 7-13
RESTART O-4
SHOW AOMABEND 17-8
SHOW AOMSTAT 17-8
SHOW CONSOLES 17-8
SHOW ISR 6-27
SHOW ISRSTATS 16-15
SHOW LSR 16-12
SHOW NCL 16-17
SHOW NDB=ALL 16-13
SHOW NP 8-12
SHOW NPTAB 8-12
SHOW PANELS 16-5
SHOW PARMS 2-2
SHOW PPIUSERS 13-15
SHOW SSIEPS 13-11
SHOW SSISTATS 13-11, 13-15
SHOW SSIUSERS 13-12
SHOW SUBSYS 15-5
SHOW SYSPARMS 2-10
SHOW SYSPGT N-2
SHOW SYSRCT N-4
SHOW SYSWAIT N-7
SHOW USERS 3-15
SHOW VSAM 16-12
SHOW VSAMIO 16-10
SSI SIGNOFF 13-12
SSI STATUS 13-12
SSI STOP 13-12
STATUS 17-8
SUBMIT 17-8
SUBS C-4
SUBSYS DEFINE 15-3
SUBSYS FORCE 15-5
SUBSYS RELOAD 15-5
SYSCMD 17-8
&INTCMD 17-4
command summary 17-8
SYSPARMS 17-4, 17-8
SYSPGT 16-17, N-9
SYSRCT 16-17, N-11

TERMINAL
dynamic DBCS specification 2-22
UDBCTL 3-22, 16-9
UDBCTL OPEN 2-23
USER ACCT 14-5
VAR C-4
VMFLKED 8-5
XNF STOP O-4
common ISR functionality 6-28
communicating between domains
INMC 6-1
overview 6-2
tuning 16-14
communication area for NCL
authorization exit 4-13, K-1
concurrent conversations (APPC) 7-3
CONIDS operand D-64
CONMSG operand D-17
CONSOLES operand D-65
control
characters, printer
adding 11-10
listing 11-11
specifying 11-9
information for user ID 3-6
members, NPF 8-6
CONUIDPW operand D-17
conversations
APPC 7-3
concurrent 7-3
security 7-16
types 7-4
ISR 6-26
correlating authorization, NCL 4-14
COS
definitions 16-4
table entry
INMC links 6-15
specifying 6-12
CPU-time accounting
implementing 14-2
starting 14-5
status 14-7
stopping 14-5
cross referencing logs 12-15
customizing
region parameters 2-5
background user 2-5

# D

DAE JCL parameter 13-6
DALDEFER operand D-17
DALRACF operand D-17
DALRLSE operand D-17
data exchange, ISR
controlling 6-28
without a procedure 6-29
data space 18-2
Data Space Manager 18-1
dumps 18-8
implementing 18-5
initialization parameters 18-5
starting 18-7
statistics 18-8
status 18-8
stopping 18-7
using 18-7
data space, areas 18-3
Dataset Access Authorization Exit (NMDSNCHK) I-1
Dataset Services Authorization Exit (NMDSSCHK) 4-20
datasets, allocating dynamically C-4
date formats 11-19
DBCS
device support
dynamic specification 2-22
enabling 2-22
identifying devices 2-22
implementation
Fujitsu 2-23
IBM 2-23
Kanji characters 2-23
dynamic terminal specification 2-22
operand D-18
support D-18
DBCS JCL parameter 2-22, 2-23, B-3
DBMS access 15-3
DD command C-4
default printers
assignment 11-14
listing 11-14
options 11-13
specifying 11-13
deferring dataset updates 16-10
DEFTERM command A-5
DEFTRANS command
LINK operand 7-11

delay interval D-15
deliver call, INMC
primary exit J-8
secondary exit J-16
delivery of VTAM messages M-1
logic M-2
DELLINK command 7-12
DELOPSET command 7-13
DESC operand D-18
devices supported by Management Services A-2
diagnosing problems
INMC links 6-13
dialogs
Initialization in Progress 2-6
System Parameters Customization 2-5
disconnect call, INMC exit J-18
DIV dataset 18-4
DLT, defining 7-12
domain ID, defining 2-3
Dynamic Link Table, defining 7-12
dynamic lists 2-7
DYNLMAX operand D-18
DYNVVOL JCL parameter B-4

# E

EASINET
activity log messages D-17
operand D-18
EDITCAPS operand D-18
EDITMAXK operand D-19
EDITNULL operand D-19
environments using ISR 6-25
EQUATE command 2-15
including in RMINIT and RMREADY 6-24
setting 2-15
EQUATES parameter group 2-14, 2-15
ERROR command C-3
errors
in activity log 12-16
in NCL authorization exit 4-12
in NPF resource tables 8-12
in structured fields G-4
EVCMDMIN operand D-19
EVERY command repeat interval D-19
exits 9-4
extended attributes support A-4
extensions to &CALL 15-3
external security packages 3-3, F-2
EXTMCS consoles and SYSCMD 17-3

# F

file transfer logs
file IDs 12-3
flushing the activity log 12-11
forcing
new password 3-8
subsystem to stop 15-5
form definitions
accessing menu 11-6
adding 11-7
listing 11-8
options 11-7
formats
activity log 12-12
logged information 12-12
PARMLIST 15-3
U28SYSIN control statements C-2
free space allocation 16-11
FTS
traffic 6-4
tuning 16-15
full security exit 3-3
function keys, customizing 2-8
functions, distributing across domains 16-3

# G

generation of SMF records 14-2
generic resources
administration utilities 5-3
GENRSRC operand D-19
global
function keys 2-8
variable prefix D-29
group ID, associating with user ID 3-6

# H

hardcopy log, format 12-13

held messages, specifying number D-20
HELDMSG operand 16-16, D-20
heterogeneous networking 7-2

# I

I (Ignore) action 2-7
ICS parameter groups
    CMDREPLS 2-14, 2-15
    EQUATES 2-14, 2-15
    for other NetMaster and SOLVE products 2-7
    FTLOGS 12-3
    SYSTEMID 2-3
ID operand 6-6, D-20
IDCAMS utility, interface to 15-3
implementation process
    customizing regions 2-5
IMS/VS, with MAI-OC 9-6
INIFILE JCL parameter B-4
INIT JCL parameter B-4
initialization
    dealing with failures 2-6
    listing failed parameter groups 2-6
    System Initialization in Progress dialog 2-6
initialization call, INMC
    primary exit J-4
    secondary exit J-14
initialization parameters
    Data Space Manager 18-5
initialization parameters, SOLVE SSI 13-5
INMC links 6-3
    BIND RUSIZE 16-14
    buffer size D-20
    controlling 6-8
    defining 6-20, 6-22
    deleting definition 6-23
    diagnosing problems 6-13
    improving performance 6-11
    ISR traffic 6-4
    listing 6-19
    logmode table 6-12
    maximum number D-18
    modifying definition 6-23
    multipath 6-3
        establishing 6-18
        multiple sessions 6-3
    planning 6-13
    preferential 6-13
        defining 6-22
        planning 6-17
    resetting 6-9
    ROF traffic 6-4
    rotate and backup 6-13
        planning 6-15
    RU size 6-12
    security exit 4-17
    simple-mode 6-13
        planning 6-14
        session parameters 6-14, 6-15
    specifying
        COS table entry 6-12
        traffic route 6-12
    starting 6-8
    static 6-5
    status 6-10
    stopping 6-9
    tracing 6-13
    traffic flow 6-4
    traffic synchronization
        preferential links 6-18
    transmission buffer size 6-11
    tuning 16-14
    using in XNF systems O-6
INMCBFSZ operand 6-11, 16-14, D-20
INMCEX01 operand D-20, J-2
INMCEX02 operand D-21, J-2
installation exits 9-4
INT JCL parameter B-4
IPAMHB operand D-21
IPCHECK operand D-21
IPPORT operand D-22
ISR
    centralized control of MS domains 6-25
    common functionality 6-28
    controlling data exchange 6-28
    conversation classes 6-26
    data exchange without a procedure 6-29
    environments that use 6-25
    establishing conversations 6-26
    message loop protection 6-26
    monitoring status 6-27
    planning 6-26
    status 6-27
    traffic 6-4
    tuning 16-15
ISTINCNO 8-5

## J


JCL
- for PPI 13-13
- for SOLVE SSI 13-3

JCL parameters B-11
- customizing Management Services 2-2
- DBCS 2-22, 2-23
- displaying current settings 2-2
- enhancing performance 16-3
- specifying 2-2
- summary table B-1

JCL parameters, specific
- AM B-3
- ARMNAME B-3
- CNM B-3
- DAE 13-6
- DBCS B-3
- DYNVVOL B-4
- INIFILE B-4
- INIT B-4
- INT B-4
- NCLFM B-5
- NCLFT B-5
- NMDID 2-3, B-5
- NMSUP B-5
- NOMODIFY B-5
- NPF B-6
- NPFFM B-6
- NPFFT B-6
- OPT B-6
- OSINP B-6
- PPO B-6
- PRI B-7
- PROD 16-3, B-7
- READY B-9
- SDUMP 13-7
- SSID B-10
- TZ B-10
- UDBDEFER B-10
- VFSENQ B-11
- VSAMIO 16-10
- WTO B-11
- XOPT B-11

JCL parameters, summary B-1

JES consoles and SYSCMD 17-2

JES2, with MAI-OC 9-6

JES3, with MAI-OC 9-7

journal swapping D-22

JRNLPROC operand D-22

JRNLSWAP operand D-22


## K


Kanji character representation 2-23


## L


LANG operand 2-20, D-22, E-2

language code 2-20, D-22

libraries, checking panel libraries 2-6

libraries, defining 3-22

LINK command 6-6

link level security (APPC) 7-16

LINK RESET command 6-9

LINK START command 7-19

LINK STAT command 6-8

LINK STOP command 6-9

LIST command C-4

lists, dynamic 2-7

LMP codes B-7

LMSGWARN operand D-22

LNKTRACE operand 6-13, D-22

lock clearing 17-4

LOCK command D-23

LOCKMAX operand D-23

LOCKPROC operand D-23

log datasets, wrapping
- hardcopy 12-15

logical unit, MAI-OC 9-4

logmode table
- definitions A-2
  - BIND A-3
  - extended attribute support A-4
  - PSERVIC field A-4
  - specifying A-6
- INMC links 6-12

logon
- attempts, number allowed D-40
- verification, security exit 4-6

LOGPAGE operand 12-13, D-23

LOGPROC operand D-23

LOGSWAP command 12-12, 12-15

LOGWRAP operand 12-15, D-23

LSR
- buffer sharing 16-9
- monitoring tuning 16-12

LU type 6.2
- option sets 7-4
- sessions 7-2

LU1 logons, alternate menu D-27
LU1BKSP operand D-23

# M

M2RETRY operand D-28
macros
  $NMNCEX1 4-13
  $NMNCPL 15-3
  OPNDST A-3
  USSMSG 8-5
MAIACBOR
  operand D-24
MAIEX02
  communications area contents L-6
    ACB open call L-8
    MAI session end call L-8
    MAI session start call L-7
    system closedown call L-6
    system initialization call L-6
  correlators
    security exit L-5
    session L-5
    system L-5
    user L-5
  description L-1
  exit
    correlators L-5
    sample L-3
    serialization L-3
  invocation L-1
  operand D-24
  re-entrancy L-2
  registers on entry L-4
  return codes L-9
    session start L-9
    system initialization L-9
  sample exit L-3
  starting L-4
  storage subpools L-2
  SYSPARMS operand L-2
MAIEX02S
  operand D-24, L-3
  sample exit L-4
MAIFILE operand D-24
MAI-OC 9-2
  on CICS/VS systems 9-5
  on IMS/VS systems 9-6
  on JES2 systems 9-6
  on JES3 systems 9-7
MAIONL operand D-25
MAIOPREF operand D-25
MAIOTRNS operand D-25
MAISREP operand 16-5
Managed Object Development Services, *see* MODS
Management Services 1-1
  connecting terminals to A-3
  connecting to SOLVE SSI 13-9
  customizing 2-1, 2-2
  identifying the system to users 2-3
  macro library 4-13
  operating environments 1-5
  regions
    domain ID 2-3
    logging on 2-5
  regionss
    defining to users 2-3
  sending query to terminal A-4
  services
    application development 1-3
    broadcast 1-3
    communication 1-3
    MODS 1-4
    NCL 1-4
    PSM 1-2
    report writer 1-3
    security 1-2
  supported devices A-2
  supported terminals 1-5
MAPDEL operand D-25
MAPFILE operand D-25
MAPLOAD operand D-25
mapped conversations (APPC) 7-4
MAPRESET operand D-26
MAXPANEL operand D-26
MAXPANL operand 16-4
MAXRUSZ operand D-26
MAXRUT2 operand D-26
MCT, defining 7-13
MDFYLIM operand D-26
MENULU1 operand D-27
MENUPROC operand D-27
message partitioning 8-3
  resource group 8-6
  resource tables
    changing status 8-13
    defining 8-8
  specifying in user ID definition 8-6
messages, maximum number
  held messages D-20
  trace messages D-30

MODDEL operand D-27
model user ID 3-9
specifying D-27
using with a partial security exit 4-5
MODETABS member A-6
MODIFY facility B-5
MODLOAD operand 4-13, 16-17, D-27
MODLUSER operand 3-9, D-27, F-11
MODS, checking control files 2-6
modules, preloading D-27
MSGCODE E-12
MSGPROC procedure 17-5
background environments 3-16
system console user ID 3-14
MSRETAIN operand D-28

# N

National Language character set support
customizing 2-20
NCL panel processing 2-21
related functions 2-21
statements
&NDBADD 2-21
&NDBPUT 2-21
&NDBSCAN 2-21
NCL 2-21
NCL
accessing user IDs 3-17
global prefix D-29
procedures
$LOBROW 12-3
$LOPROC 12-3
$NDJPROC D-22
$NMLOCK D-23
$NMPMENU D-27
banner exit $PSBA80X 11-20
improving procedure use 16-7
INIT member 2-2
monitoring procedure loading 16-7
NET/MAIL exit
$PSML78X 11-20
$PSML79X 11-20
preloading 16-8
PSM to dataset exit
$PSDS81X 11-28
READY member 2-2
sharing 16-7
specifying storage limit 16-8
SYSCMD 17-4
unloading D-50
using
in XNF systems O-8
with ROF 6-25
verb summary for SYSCMD 17-9
NCL authorization exit 4-11
activating 4-12
and $CACNTL 4-12
and $PSPOOL 4-12
communication area 4-13, K-1
NEXCORR field 4-14
NEXUTOKN field 4-14
controlling access levels 4-11
correlating authorization 4-14
designing 4-16
errors 4-12
preloading 4-13
providing additional checking 4-13
sample 4-17
specifying exit name D-28
SQL database access 4-11
UDB access 4-11, 4-14
from NCL procedures 4-15
writing SMF records 4-11, 4-16
NCLCSAVE operand D-28
NCLEX01 operand 4-12, D-28
NCLFM JCL parameter B-5
NCLFT JCL parameter B-5
NCLGBTRC operand D-28
NCLGLBL operand D-29
NCLMAXK operand D-29
NCLOGTRM operand D-29
NCLPEND operand D-30
NCLPRSHR operand 16-7, D-30
NCLTRLFF operand D-30
NCLTRMAX operand D-30
NCLUMAX operand D-31
NCLXUSER operand D-31
NDB, tuning 16-9
database activity 16-11
record sizes 16-11
NDBDIOL operand D-31
NDBDRCL operand D-31
NDBDSTL operand D-31
NDBDTML operand D-32
NDBLOGSZ operand D-34
NDBMIOL operand D-32
NDBMRCL operand D-33
NDBMSTL operand D-33
NDBMTML operand D-33

NDBOPENX operand D-34
NDBPHONX operand D-34
NDBSCANO operand D-32
NDBSUBMN operand 16-11, D-34
NDBSUBMX operand 16-11, D-34
NetMaster for File Transfer regions, customizing 2-5
NetMaster, NetSpy, and SOLVE products 1-5
NetSpy
  alerts 19-3
  commands 19-5
  connections to NetMaster 19-2
  ICS parameters 19-7
  SNA agents 19-1
  user alert monitors 19-3
Network Control Language, *see* NCL
Network Partitioning Facility, see NPF
networking, heterogeneous 7-2
NEWS, SYSPARMS operands D-53
NEXCORR field 4-14
NEXUTOKN field 4-14
NMDID JCL parameter 2-3, B-5
NMDSNCHK exit I-1
NMDSSCHK exit 4-20
NMIQLIM operand D-35
NMISTCMM module 8-5
NMMACLIB file 4-13
NMREADY 6-6
NMSUP JCL parameter B-5
NOMODIFY JCL parameter B-5
NONSWAP operand D-35
notify call, INMC exit J-11
NPF 8-2
  command
    members 8-4, 8-6
    partitioning 8-2
  control members 8-6
  dataset 8-4
  JCL parameter B-6
  library 8-6, 8-9
  message
    members 8-6
    partitioning 8-2
  resource groups 8-6
  resource list member 8-6
  resource tables
    changing status 8-13
    coding 8-9
    defining 8-8
    deleting 8-12
    displaying 8-13
    errors 8-12
    loading 8-11
    replacing online 8-12
    testing 8-12
  storing partitioning information 8-4
  using
    with ROF 8-14
    with system console 8-14
  VTAM message tables 8-5
NPFFM JCL parameter B-6
NPFFT JCL parameter B-6
NPFFT startup parameter 8-4
NPTAB command 8-3, 8-11
NPTABLES DD statement 8-4
NRDLIM operand D-35
NSPRSPTO operand D-36
NSPRTINT operand D-35
NTS, SYSPARMS operands D-54
NTSACBNM operand D-54
NTSACCT operand D-54
NTSCINTV operand D-55
NTSCLOSE operand D-55
NTSCNMQ operand D-55
NTSEVENT operand D-55
NTSINTSV operand D-56
NTSMAIEX operand D-56
NTSMAISV operand D-56
NTSMAXTP operand D-56
NTSMAXTR operand D-56
NTSRSINT operand D-56
NTSRSLIM operand D-57
NTSRSTAT operand D-56
NTSSAWBF operand D-57
NTSSKEEP operand D-57
NTSTRBFX operand D-57
NTSTRCBF operand D-57

# O

OCS
  command authority levels H-1
  command stack size D-16
OCSHLITE operand D-36
OCST, defining 7-13
OCSTIME operand D-36

OMLUMAX operand D-36
online activity log 12-12
operating environments 1-5
Operator Console Services, *see* OCS
OPNDST macro A-3
OPT JCL parameter B-6
option sets, LU6.2 (APPC) 7-4
OSHOST operand D-36
OSINP JCL parameter B-6

# P

panel
- definitions 3-22
- library 3-22
- number held in storage D-26
- paths 3-22
- tuning 16-4, 16-5

panel libraries, checking 2-6
PANLBFSZ operand 16-5, D-37
PANLBUFF operand 16-5, D-37
PANLPPD operand D-37
paper definitions
- adding 11-7
- listing 11-8
- options 11-7

parallel sessions, links 7-15
PARMSEP command C-3
partial security exit, model user IDs 4-5
passwords
- expiry interval D-40
- forcing change 3-8
- length D-40

paths, defining 3-22
peer protocol 7-2
performance
- considerations when writing NCL 16-16
- distributing functions across domains 16-3
- enhancing 16-1
  - by controlling OCS message flow 16-16
  - by deferring VSAM dataset updates 16-10
  - by performing VSAM I/O in subtask 16-10
  - by tuning communication between domains 16-14
  - by tuning record size 16-11
  - by using buffer sharing 16-9
  - by using COS definitions 16-4
- improving NCL procedure use 16-7
- monitoring NCL procedure loading activity 16-7
- NCL procedures
  - preloading 16-8
  - sharing 16-7
- tuning
  - access to panels 16-5
  - at the system level 16-2
  - dataset activity 16-11
  - panel use 16-4
  - storage limits for panel sends 16-5
  - VTAM interface 16-4

performance and tuning commands N-1
- SHOW SYSPGT N-2
- SHOW SYSRCT N-4
- SHOW SYSWAIT N-7
- SYSPGT N-9
- SYSRCT N-11

PF command 2-8
PF LIST command 2-8
PGM command C-3
physical terminal 9-4
PPI
- activating 13-14
- executing commands 13-15
- implementing 13-12
- JCL 13-13
- using with SOLVE SSI 13-12

PPO JCL parameter B-6
PPOACBNM operand D-37
PPOCOLOR operand D-37
PPOHLITE operand D-37
PPOPROC
- operand D-38
- procedure 6-28

PPOSOCMD operand D-38
PPOSOMSG operand D-38
PPOUSMSG operand D-39, M-3
PPREF command C-3
preferential INMC links 6-13
- defining 6-22
- planning 6-17
- traffic synchronization 6-18

PRELOAD operand D-39
preloading NCL procedures 16-8, D-51
preparse facilities 16-16

PRI JCL parameter B-7
PRIACBNM operand D-40
primary menu
alternate D-27
procedure 3-18
Print Services Manager, *see* PSM
print spool file 12-11
printer definitions
adding 11-4
administering 11-16
assigning 11-3
listing 11-5
maintaining 11-16
menu 11-3
printer exit procedure
for writing to dataset 11-28
for writing to NET/MAIL 11-20
printer requirements
clearing printer spool 11-18
control characters
listing 11-11
specifying 11-9
default printers 11-13
paper definitions 11-6
setup definition 11-10
printer spool 11-18, 11-19
PROD JCL parameter 16-3, B-7
PROFILE command D-20, E-12, E-13
PSERVIC field, logmode table definition A-4
PSM
accessing 11-3
banner exit $PSBA80X 11-20
concepts
defaults 11-16
setup 11-9
customizing 11-2
date formats 11-19
default printers 11-13
facilities 11-2
printer definitions 11-3
printer exit for dataset 11-28
$PSDS81X procedure 11-28
$PSDS81Z procedure 11-28
exit parameters 11-28
processing 11-28
PSM NET/MAIL printer exit 11-20
output
formatting 11-22
segmentation 11-21
parameters 11-22
$PSML78X 11-22
$PSML79X 11-22
CLASS 11-23
DOMAIN 11-24
LINK 11-24
LU 11-24
MAILID 11-22
PAGE 11-25
PRIVATE 11-23
PROFILE 11-24
SKIP 11-24
SUBJECT 11-23
USCORE 11-25
VIEWONLY 11-23
prerequisites 11-20
remote access 11-21
sample definitions 11-26
source of mail 11-21
PSM panels
Administration Menu 11-16
Clear Spool 11-18
Default Definition 11-17
Default Printer Assignment 11-14
Default Printer Assignment List 11-15
Default Printer Assignment Menu 11-13
Form Definition 11-8
Form Definition List 11-9
Form Definition Menu 11-7
Printer Definition 11-5
Printer Definition List 11-6
Printer Definition Menu 11-4
Setup Definition 11-11
Setup Definition List 11-12
Setup Definition Menu 11-10
PSUFF command C-3
PWEXPIRE operand D-40, E-5
PWMAX operand D-40, E-1
PWMIN operand D-40, E-1
PWRETRY operand D-40

# Q

query, extended attributes A-4
queuing limits for message traffic D-35

# R

RACF user ID, signing on 3-13
READY JCL parameter B-9

RECONN operand D-40
RECSZ 16-11
region use (APPC) 7-19
registers, INMC security exit J-3
REPLINK command 7-12
REPOPSET command 7-13
requests
  &CALL 15-2
  unit size D-26
resource definitions
  listing 10-3
  types of resources 10-5
resource group
  command partitioning 8-6
  message partitioning 8-6
resource list member 8-6
resource tables
  coding 8-9
  defining 8-8
  deleting 8-12
  displaying 8-13
  errors 8-12
  loading 8-11
  replacing online 8-12
  testing 8-12
RESTART command O-4
retry interval, SOLVE SSI 13-10
return codes, &SMFWRITE 4-16
RMINIT with EQUATE command 6-24
RMREADY with EQUATE command 6-24
ROF
  traffic 6-4
  tuning 16-15
  user IDs 6-24
  using
    for centralized control 6-24
    with background environments 3-16, 6-25
    with NCL procedures 6-25
    with NPF 8-14
    with system console 3-14, 6-25
ROFPAREN operand D-40
ROFPREFX operand D-41
rotate and backup INMC links 6-13
  planning 6-15
  traffic synchronization 6-18
ROUTCDE operand 3-14, D-41
RTA480 operand D-41
RU size, increasing 6-12

# S

SAF user token 4-14
sample MAIEX02 exit L-3
screen sizes
  specifying defaults D-53
  specifying size A-4
  unspecified viewport size A-4
SDUMP JCL parameter 13-7
security 3-2
  for APPC links 7-16
  implementing 3-1
    additional 4-1
    for the first time 3-5
    in XNF systems O-6
  UAMS 3-2
security exit 4-2
  ACF2security exit
    SAF partial exit 3-3
  full 3-3
  INMC 4-17
    changing name dynamically J-2
    execution J-1
    primary exit 4-17
      calls 4-18
      deliver call J-8
      initialization call J-4
      linking to MS J-2
      notify call J-11
      writing J-3
    registers J-3
    secondary exit 4-17, 4-18
      calls 4-18
      deliver call J-16
      disconnect call J-18
      initialization call J-14
      linking to MS J-2
      writing J-14
  partial, using model user IDs 4-5
  RACF 3-3
  user ID 4-2
    adding definitions 4-10
    adding new functionality 4-8
    calls F-6
    changing passwords 4-7
    controlling access to system 4-3
    deleting definitions 4-10
    execution F-5
    full 4-2
    functions 4-2
    listing definitions 4-10
    logon verification 4-6
    partial 4-2

retrieving definitions 4-7
updating definitions 4-8
writing F-3
sequence of structured fields G-4
services
Management Services
application development 1-3
broadcast 1-3
communication 1-3
PSM 1-2
report writer 1-3
security 1-2
session establishment (APPC) 7-22
sessions
APPC 7-3
multiple 6-3
with other systems 9-4
SESSMSG operand 6-13, D-41
setup definition
adding 11-10
listing 11-11
menu 11-9
severity, VTAM messages M-1
sharing NCL procedures 16-7
SHOW AOMABEND command 17-8
SHOW AOMSTAT command 17-8
SHOW CONSOLES command 17-8
SHOW ISR command 6-27
SHOW ISRSTATS command 16-15
SHOW LSR command 16-12
SHOW NCL command 16-17
SHOW NDB=ALL command 16-13
SHOW NP command 8-12
SHOW NPTAB command 8-12
SHOW PANELS command 16-5
SHOW PARMS command 2-2
SHOW PPIUSERS command 13-15
SHOW SSIEPS command 13-11
SHOW SSISTATS command 13-11, 13-15
SHOW SSIUSERS command 13-12
SHOW SUBSYS command 15-5
SHOW SYSPARMS command 2-10
SHOW SYSPGT command N-2
SHOW SYSPGTcommand N-2
SHOW SYSRCT command N-4
SHOW SYSWAIT command N-7
SHOW USERS command 3-15
SHOW VSAM command 16-12
SHOW VSAMIO command 16-10
simple-mode INMC links 6-13
COS table entry 6-15
planning 6-14
session parameters 6-14, 6-15
single session links 7-14
SMF records 14-2
controlling access to write 4-11, 4-16
generation 14-2
identifier D-42
SMFID operand D-42
SOLVE products 1-5
SOLVE SSI 13-2, 18-3
activating 13-9
executing commands 13-11
implementing 13-1, 13-2
initialization parameters 13-5
JCL 13-3
retry interval 13-10
terminating 13-10
using with PPI 13-12
SOLVE subsystem interface
*see SOLVE SSI*
SQL databases, controlling access 4-11
SSI SIGNOFF command 13-12
SSI STATUS command 13-12
SSI STOP command 13-12
SSID JCL parameter B-10
SSID operand 13-9, D-42
SSIRTINT operand 13-10, D-42
static INMC links 6-5
statistical information for user ID 3-6
STATUS command 17-8
status of CPU-time accounting 14-7
STGWRN operand D-42
STGWRNXA operand D-42
storage limits for panel sends 16-5
storage, maximum amount for editor D-19
structured fields G-1
descriptions G-4
error conditions G-4
sequence G-4
structured fields, specific
AOM message delivery and routing codes G-29
AOM MVS SYSCMD console authority G-30
AOM user ID MSG level G-31
AOM user ID VM SYSCMD

authority G-32
AOM user ID VOS3/JSS4 SYSCMD command authority G-32
APPC access key G-22
APPC access lock G-23
broadcast services access G-10
CAS command maintenance access G-51
CAS criteria maintenance access G-51
CAS help maintenance access G-50
CAS list maintenance access G-49
CAS menu maintenance access G-49
CAS message maintenance G-50
CAS panel domain maintenance access G-49
CAS panel maintenance access G-12
CAS table maintenance access G-50
date/time user ID last updated G-8
FTS access G-12
group definition G-9
INFO/MASTER maintenance access G-26
information services access G-27
initial OCS command G-18
installation attribute 5 G-24
installation attribute field 1 G-23
installation attribute field 10 G-26
installation attribute field 2 G-23
installation attribute field 3 G-24
installation attribute field 4 G-24
installation attribute field 6 G-25
installation attribute field 7 G-25
installation attribute field 8 G-25
installation attribute field 9 G-26
library services path name G-16
MAI-FS A and E command capability G-33
MAI-FS access G-13
MAI-FS model user ID G-33
MAI-FS privilege class G-33
Management Services access G-37
Mapping Services access G-48
message code value G-19
MODS access G-46
MODS administration access G-46
MODS application register access G-46
MSGPROC member G-17
multiple signon capability G-8
NCL procedure library G-14
NCPView access G-27
NCS access G-15
network services access G-11
NEWS access G-12
NEWS reset G-28
notification details (first rule) G-38
notification details (fourth rule) G-41
notification details (second rule) G-39
notification details (third rule) G-40
NPF command member G-17
NPF message restriction G-19
NTS access G-28
object services access G-47
object services object class specification access G-47
object services security access G-48
OCS access G-10
OCS command authority level G-16
OCS monitor status G-17
OCS MSG message receipt G-20
OCS unsolicited message receipt G-20
operations management access G-14
panel command access G-36
password expiry action indicator G-10
PPO message receipt G-18
PPO message severity level G-18
PSM ability to change print request priority G-35
PSM administration access G-34
PSM maintenance access G-34
PSM primary menu access G-34
PSM queue access for all output G-35
PSM queue access for own output G-36
Report Writer administration access G-43
Report Writer maintenance access G-43
Report Writer primary menu access G-42
Report Writer private report access for all reports G-44
Report Writer private report access to own reports G-44
Report Writer public report access G-43
Report Writer schedule maintenance access G-45
resource list member G-20
SOLVE:Asset primary menu ID G-53
SOLVE:Asset security profile G-53
SOLVE:Attach for SNMP access G-42

SOLVE:Central access G-45
SOLVE:Central primary menu G-47
SOLVE:Central security profile G-48
SOLVE:Change primary menu ID G-52
SOLVE:Change security profile G-52
SOLVE:Configuration primary menu ID G-54
SOLVE:Configuration security profile G-53
SOLVE:Helpdesk primary menu ID G-55
SOLVE:Helpdesk security profile G-54
SOLVE:NetMail access G-45
SOLVE:Problem primary menu ID G-52
SOLVE:Problem security profile G-51
SPLIT/SWAP access G-15
system support access G-11
terminal restrictions G-7
timeout control G-7
TSO autologon access G-15
UAMS access G-14
user ID name G-5
user ID suspend date G-6
user ID type G-9
user language code G-6
user location G-5
user name G-5
user services procedure name G-13
user telephone number G-6
user time zone G-16
user timeout action (1) G-21
user timeout action (2) G-22
user timeout period (1) G-21
user timeout period (2) G-21
SUBMIT command 17-8
SUBS command C-4
SUBSYS DEFINE command 15-3
SUBSYS facility 15-2
PARMLIST format 15-3
reloading the program 15-5
sending parameter lists 15-3
subsystems
defining 15-4
deleting 15-5
displaying status 15-6
starting 15-4
stopping 15-5
by force 15-5
using 15-1, 15-3
writing a program 15-6
in high-level languages 15-8
SUBSYS FORCE command 15-5
SUBSYS RELOAD command 15-5
subsystems
defining 15-4
deleting 15-5
displaying status 15-6
reloading the program 15-5
sending parameter lists 15-3
starting 15-4
stopping 15-5
by force 15-5
supported terminals 1-5
SYSCMD
AOM command summary 17-8
facility 17-1
message prefixes 17-4
NCL verbs summary 17-9
SYSPARMS operands D-58
variables summary 17-9
SYSCMD command 17-8
SYSCONNM operand D-43
SYSCONSO operand D-43
SYSCONUI operand 3-13, D-43, D-44
SYSCONXU operand D-44
SYSLOG operand 12-16, D-44
SYSLOGFM operand D-46
SYSOUT 12-15
SYSPARMS command 17-4, 17-8
SYSPARMS operands
for MAI-OC 9-3
for NEWS D-53
for NTS D-54
for SYSCMD D-58
generic D-12
SYSPARMS, general information
changing authority level 2-11
changing settings 2-10
command format 2-10
customizing MAI-OC 9-3
displaying current settings 2-10
for other MS products 2-12
restricting use of 2-11
specifying in INIT member 2-11
SYSPARMS, generic operands
ABENDCD operand D-12
ACBRETRY operand D-12
APPLSTAT operand D-12
APPLSTIV operand D-13

APPTXT1 operand D-13
APPTXT2 operand D-13
APPTXT3 operand D-13
APPTXT4 operand D-14
APPTXT5 operand D-14
APPTXT6 operand D-14
AUTOEXEC operand D-14
CALLSHR0 operand D-15
CDELAY operand D-15
CMDAUTH operand 2-12, 2-13, D-15, H-2
CMDREPL operand D-16
CMDSTACK operand D-16
CONMSG operand D-17
CONUIDPW operand D-17
DALDEFER operand D-17
DALRACF operand D-17
DALRLSE operand D-17
DBCS operand D-18
DESC operand D-18
DYNLMAX operand D-18
EASINET operand D-18
EDITCAPS operand D-18
EDITMAXK operand D-19
EDITNULL operand D-19
EVCMDMIN operand D-19
GENRSRC operand D-19
HELDMSG operand 16-16, D-20
ID operand 6-6, D-20
INMCBFSZ operand 6-11, 16-14, D-20
INMCEX01 operand D-20, J-2
INMCEX02 operand D-21, J-2
IPAMHB operand D-21
IPCHECK operand D-21
IPPORT operand D-22
JRNLPROC operand D-22
JRNLSWAP operand D-22
LANG operand 2-20, D-22, E-2
LMSGWARN operand D-22
LNKTRACE operand 6-13, D-22
LOCKMAX operand D-23
LOCKPROC operand D-23
LOGPAGE operand 12-13, D-23
LOGPROC operand D-23
LOGWRAP operand 12-15, D-23
LU1BKSP operand D-23
M2RETRY operand D-28
MAIACBOR operand D-24
MAIEX02 operand D-24
MAIEX02S operand D-24, L-3
MAIFILE operand D-24
MAIONL operand D-25
MAIOPREF operand D-25
MAIOTRNS operand D-25
MAISREP operand 16-5
MAPDEL operand D-25
MAPFILE operand D-25
MAPLOAD operand D-25
MAPRESET operand D-26
MAXPANEL operand D-26
MAXPANL operand 16-4
MAXRUSZ operand D-26
MAXRUT2 operand D-26
MDFYLIM operand D-26
MENULU1 operand D-27
MENUPROC operand D-27
MODDEL operand D-27
MODLOAD operand 4-13, 16-17, D-27
MODLUSER operand 3-9, D-27, F-11
MSRETAIN operand D-28
NCLCSAVE operand D-28
NCLEX01 operand 4-12, D-28
NCLGBTRC operand D-28
NCLGLBL operand D-29
NCLMAXK operand D-29
NCLOGTRM operand D-29
NCLPEND operand D-30
NCLPRSHR operand 16-7, D-30
NCLTRLFF operand D-30
NCLTRMAX operand D-30
NCLUMAX operand D-31
NCLXUSER operand D-31
NDBDIOL operand D-31
NDBDRCL operand D-31
NDBDSTL operand D-31
NDBDTML operand D-32
NDBLOGSZ operand D-34
NDBMIOL operand D-32
NDBMRCL operand D-33
NDBMSTL operand D-33
NDBMTML operand D-33
NDBOPENX operand D-34
NDBPHONX operand D-34
NDBSCANO operand D-32
NDBSUBMN operand 16-11, D-34
NDBSUBMX operand 16-11, D-34
NMIQLIM operand D-35
NONSWAP operand D-35
NRDLIM operand D-35
NSPRSPTO operand D-36
NSPRTINT operand D-35
OCSHLITE operand D-36
OCSTIME operand D-36
OMLUMAX operand D-36
OSHOST operand D-36

PANLBFSZ operand 16-5, D-37
PANLBUFF operand 16-5, D-37
PANLPPD operand D-37
PPOACBNM operand D-37
PPOCOLOR operand D-37
PPOHLITE operand D-37
PPOPROC operand D-38
PPOSOCMD operand D-38
PPOSOMSG operand D-38
PPOUSMSG operand D-39, M-3
PRELOAD operand D-39
PRIACBNM operand D-40
PWEXPIRE operand D-40, E-5
PWMAX operand D-40, E-1
PWMIN operand D-40, E-1
PWRETRY operand D-40
RECONN operand D-40
ROFPAREN operand D-40
ROFPREFX operand D-41
ROUTCDE operand 3-14, D-41
RTA480 operand D-41
SESSMSG operand 6-13, D-41
SMFID operand D-42
SSID operand 13-9, D-42
SSIRTINT operand 13-10, D-42
STGWRN operand D-42
STGWRNXA operand D-42
SYSCONNM operand D-43
SYSCONSO operand D-43
SYSCONUI operand 3-13, D-43, D-44
SYSCONXU operand D-44
SYSLOG operand 12-16, D-44
SYSLOGFM operand D-46
SYSUDB operand D-46
TCMDMAX operand D-46
TELNET operand D-46
TNDSREG operand D-46
TNGPREFX operand D-47
TNSPREFX operand D-47
TOACT1 operand D-48
TOACT2 operand D-48
TOALARM operand D-48
TODISC operand D-48
TOLOCK operand D-48
TOSTART operand D-48
TOTIME1 operand D-49
TOTIME2 operand D-49
TPEND operand D-49
TRACEOPT operand D-50
UNLOAD operand D-16, D-50
UNPVPSZ operand D-51
UNSPVPSZ operand A-5
USERPW operand D-51
VDISPLAY operand 8-3, D-51
VSAMCAT operand D-52
VSAMMODE operand D-52
VTAMAUTH operand D-52
VTAMCDRM operand D-52
VTAMID operand D-52
VTAMNTFY operand D-52
XABELOW operand D-53
XNFRETRY operand D-53
XNFSCRSZ operand D-53, O-3
XNFUCENM operand D-53

SYSPARMS, NEWS-specific operands
CNMACBNM operand D-53
CNMPROC operand D-53

SYSPARMS, NTS-specific operands
NTSACBNM operand D-54
NTSACCT operand D-54
NTSCINTV operand D-55
NTSCLOSE operand D-55
NTSCNMQ operand D-55
NTSEVENT operand D-55
NTSINTSV operand D-56
NTSMAIEX operand D-56
NTSMAISV operand D-56
NTSMAXTP operand D-56
NTSMAXTR operand D-56
NTSRSINT operand D-56
NTSRSLIM operand D-57
NTSRSTAT operand D-56
NTSSAWBF operand D-57
NTSSKEEP operand D-57
NTSTRBFX operand D-57
NTSTRCBF operand D-57

SYSPARMS, SYSCMD-specific operands
AOMCMIGI operand D-59
AOMCMIGX operand D-59
AOMCOPTS operand D-59
AOMCPREF operand D-60
AOMCTO1 operand D-60
AOMCTO2 operand D-60
AOMCTO3 operand D-60
AOMCTYPE operand D-61
AOMCUTOK operand D-61
AOMGQSZ operand D-62
AOMJESCH operand D-62
AOMMLTO operand D-62
AOMPQSZ operand D-62
AOMPRFJI operand D-63
AOMPRFJN operand D-63
AOMPRFMN operand D-63
AOMPRFSN operand D-63
AOMPRFTM operand D-63

AOMSSID operand D-63
AOMSUBCH operand D-64
AOMTRACE operand D-64
AOMTRCRC operand D-64
AOMTRLIM operand D-64
CONIDS operand D-64
CONSOLES operand D-65
SYSPGT command 16-17, N-9
SYSPLEX environment C-1
SYSRCT command 16-17, N-11
system console
command replies 3-13
defining user ID 3-10
MVS/ESA 4.1 or later 3-10
VM 3-13
MSGPROC procedure 3-14
unsolicited output 3-14
using
with NPF 8-14
with ROF 3-14, 6-25
system groups
accessing 10-2
adding 10-3
listing 10-2
resource definitions 10-4
system identifier 2-3, D-20
system log 12-16
PPO messages 12-16
system parameters
initial customization dialog 2-5
listing parameter groups that failed initialization 2-6
parameter group lists 2-7
system performance, enhancing 16-2
system procedures, background 3-15
system variables
&FILERC 4-11
&USERPW D-51
&ZAMTYPE O-8
&ZDOMID 17-7
&ZMAOMAU 17-9
&ZMPTEXT 17-5
&ZUCENAME O-8
&ZUSERLC E-2
SYSTEMID parameter 2-3
SYSUDB operand D-46

# T

TCMDMAX operand D-46
TCT, defining 7-9
TELNET operand D-46
TERMINAL command, DBCS 2-22
terminal timeout
actions 2-17
changing parameters 2-19
considerations 2-20
implementing timeout facility 2-19
setting parameters 2-15
first level 2-16
second level 2-17
starting timeout facility 2-19
trigger event 2-18
terminals
connecting to Management Services A-3
specifying screen sizes A-4
unspecified viewport size A-4
timeout
actions D-48
APPC 7-24
changing parameters 2-19
implementing 2-19
intervals D-49
trigger event 2-18
timer commands 12-13
TNDSREG operand D-46
TNGPREFX operand D-47
TNSPREFX operand D-47
TOACT1 operand D-48
TOACT2 operand D-48
TOALARM operand D-48
TODISC operand D-48
TOLOCK operand D-48
TOSTART operand D-48
TOTIME1 operand D-49
TOTIME2 operand D-49
TPEND operand D-49
TPN, specifying 7-10
trace messages, maximum number D-30
TRACEOPT operand D-50
tracing
INMC links 6-13
options D-50
traffic
flow, INMC 6-4
synchronization, INMC 6-18
transaction programs 7-2
transmission buffer size 6-11

TZ JCL parameter B-10

# U

U28SYSIN dataset C-2
U29SYSIN dataset C-2
UAMS 3-2, E-1
UAMS panels
    Access Authorities E-7
    Installation Attributes E-11
    MODS Details E-10
    Network Management Details E-16
    Notification Details E-14
    OCS Details E-11
    Print Services Manager Details E-17
    Report Writer Details E-19
    User Authorities E-4
    User Details 3-8, E-1
UCE O-4
UDB
    access from NCL procedures 4-15
    controlling access 4-11
    tuning 16-9
UDBCTL command 3-22, 16-9
UDBCTL OPEN command 2-23
UDBDEFER JCL parameter B-10
ULE O-4
UNLOAD operand D-16, D-50
UNPVPSZ operand D-51
unsolicited VTAM message delivery M-1
    delivery logic M-2
unspecified viewport size A-4
    specifying D-51
UNSPVPSZ operand A-5
user abend code, specifying D-12
USER ACCT command 14-5
user data, accepting at logon D-17
user exits, migration IDs P-1
user ID
    accessing from NCL 3-17
    associating with group ID 3-6
    control information 3-6
    defining 3-6
        background environment 3-15
        for ROF 6-24
    listing defined 3-7
    model 3-9
    RACF, signing on 3-13
    security exit
        see security exit, user ID
    statistical information 3-6
USERPW operand D-51
users, defining to the system 3-5
USSMSG macros 8-5
UTIL0028 utility C-1
    commands C-3
    processing C-7
    using C-2
UTIL0035 utility 15-3
UTOKEN 4-14

# V

VAR command C-4
variables, *see system variables*
VDISPLAY operand 8-3, D-51
verbs
    &DOM 17-9
    &INTCMD 6-25, 12-13, E-13
    &INTREAD 6-25
    &LOCK 15-2, D-23
    &LOGCONT 12-3
    &PANEL 3-22
    &RETCODE 17-10
    &SECCALL E-11, F-18
    &SMFWRITE D-28
    &WRITE E-12
    &WTO 17-9
    &WTOR 17-9
    &ZDOMID 17-10
    &ZFDBK 17-10
    &ZMAOMBC 17-9
    &ZMAOMDATA 17-9
    &ZMAOMID 17-9
    &ZMAOMJI 17-9
    &ZMAOMJN 17-9
    &ZMAOMMID 17-9
    &ZMAOMMIN 17-9
    &ZMAOMMLC 17-9
    &ZMAOMMLD 17-9
    &ZMAOMMLE 17-9
    &ZMAOMMLL 17-10
    &ZMAOMMLT 17-10
    &ZMAOMMLV 17-10
    &ZMAOMMSG 17-10
    &ZMAOMRC 17-10
    &ZMAOMRCM 17-10
    &ZMAOMRCX 17-10
    &ZMAOMSOS 17-10
    &ZMAOMTM 17-10

&ZMAOMTYP 17-10
&ZMAOMUFM 17-10
VFS database 6-20, 6-21, 6-22
VFSENQ JCL parameter B-11
virtual
routes for INMC links 6-12
VMFLKED command 8-5
VSAM datasets
deferring updates 16-10
I/O in subtask 16-10
sharing buffers 16-9
tuning 16-9
activity 16-11
monitoring 16-12
monitoring activity 16-13
size 16-11
VSAMCAT operand D-52
VSAMIO JCL parameter 16-10, B-11
VSAMMODE operand D-52
VTAM
ACB, reopening D-12
APPL
definition, checking status D-12
name, changing dynamically D-40
logmode table
extended attributes A-4
message table requirements 8-5
messages
delivery M-1
logic M-2
severity M-1
PPO ACB name, changing D-37
sessions 6-3
starting D-52
VTAM display command 8-3
VTAMAUTH operand D-52
VTAMCDRM operand D-52
VTAMID operand D-52
VTAMNTFY operand D-52

# W

warning for lost messages D-22
wrapping log datasets
hardcopy 12-15
writing
INMC security exits
primary J-3
secondary J-14
NCL, performance considerations 16-16
SUBSYS programs 15-6
in high-level languages 15-8
user ID security exit F-3
WTO JCL parameter B-11

# X

XABELOW operand D-53
XNF product O-2
connecting terminals to Management Services O-3
customizing Management Services O-4
implementing security O-6
issuing Management Services commands O-7
restarting system O-4
shutting down system O-4
using
APPC links O-7
for Management Services O-1
INMC links O-6
NCL components O-8
XNF STOP command O-4
XNFRETRY operand D-53
XNFSCRSZ operand D-53, O-3
XNFUCENM operand D-53
XOPT JCL parameter B-11