



IBM Session Manager for z/OS

Installation and Customization

Version 3 Release 1



IBM Session Manager for z/OS

Installation and Customization

Version 3 Release 1

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 477.

This edition applies to Version 3 Release 1 of IBM Session Manager for z/OS, program number 5655-U98, and to all subsequent versions, releases, and modifications until otherwise indicated in new editions.

Copyright © 2003-2011 All Rights Reserved. Macro 4 Limited - a division of UNICOM Systems, Inc.

Contents

About this manual	15
Session Manager documentation	17
Conventions	18
New features	19
New in Version 3.100 (Session Manager Release)	19
New in Version 2.2.05 (IBM PTF: UK55397, APAR: PM05796)	26
New in Version 2.2.00 (Session Manager Release)	31
New in Version 2.1.05 (IBM PTF: UK37522, APAR: PK65596)	36
New in Version 2.1.00 (Session Manager Release)	36
New in Version 1.3.15 (IBM PTF: UK20403, APAR: PK33701)	39
New in Version 1.3.10 (IBM PTF: UK15956, APAR: PK21439)	43
New in Version 1.3.05 (IBM PTF: UK11656, APAR: PK16122)	45
New in Version 1.3.00 (Session Manager Release)	48
Chapter 1 Deciding on a Classic or an OLA system	49
Overview	50
Frequently-asked OLA questions	51
Batch Administration	55
Further information	56
Chapter 2 Performing a basic Classic or OLA installation	57
Product changes – Session Manager 2.2.00 and higher	58
Overview	58
Product changes from 2.200 onwards	58
Prerequisites	61
Installing a basic Classic or OLA system	62
Step 1: Preliminary checks	62
Step 2: Installation	62
Step 3: APF authorization	62
Step 4: VTAM alterations	63
Step 5: Configuration file customization	65

Step 6: Starting the system – initiating Session Manager	66
Step 7: Signing on to Session Manager	67
Chapter 3 Installing a Functional Enhancement PTF	71
Introduction	72
Applying a PTF in a Classic system	73
Installing the PTF	73
Customizing Session Manager	73
Putting Session Manager into production	74
Applying a PTF to an OLA system	75
Installing the PTF	75
Customizing Session Manager	75
Putting Session Manager into production	76
Chapter 4 Upgrading to new version or release	77
Upgrading Classic to Classic	78
Installing Session Manager	78
Customizing Session Manager	78
Putting Session Manager into production	80
Upgrading OLA-enabled to OLA-enabled	81
Installing Session Manager	81
Customizing Session Manager	81
Putting Session Manager into production	83
Chapter 5 Post-installation configuration issues	85
Storage requirements	86
Estimating the dynamic storage requirement	86
Examples of storage requirements	87
Specifying the size required	88
Handling upgrades and Functional Enhancement PTFs	89
Supplied configuration samples	89
Supplied exit samples	95
Supplied JCL samples	97
Implementing security	99
Accessing applications	100
Choosing an ACB	100
Further information	102
Defining System, Profile and User settings	103
Using profiles	103
Choosing the configuration statements	103
Common enduser parameters	105
Common session parameters	106
Determining Profile order	107
Setting authorization levels and classes	107
Flexible application access	107
Priority sessions	108
Configuration statement syntax	108

Examples	108
Designing panels and menus	111
Panel sections	112
Panel sub-definitions	113
Basic field types	113
Further information	114
National language support	115
Adding/ customizing LPs	115
Specifying the language for screen text	116
Additional customer-defined messages and panels	116
Restrictions	116
Summary	117
Further information	117
Using scripts with applications	118
Session scripts	118
Exit scripts	119
Command scripts	120
Windows scripts	120
Application builder scripts	121
Understanding execution of a script	121
Session Manager variables	121
Further information	121
Summary	122
User Exits and the Multiple Exit Driver	124
Setting TCP/IP support (if required)	125
Enabling Eclipse support	126
Networking and Sysplex considerations (if required)	127
Setting Audit GDG dataset support (if required)	133
Setting Dump GDG Dataset Support (if required)	133
Implementing Automatic Restart Manager (if required)	135
Defining an ARM policy	135
Modifying the Session Manager startup JCL	135
Setting up the security definitions	135
Using multiple Session Manager instances – networking	137
Requirements	137
Nodes	137
Routing	138
Internal names in the network	138
Sysplex networking	138
VTAM implications	139
Using Online Administration (OLA)	139
Using the VTM facility	139
Further information	140
Setting up OLA for single and multiple Session Manager instances	141
Administering a single Session Manager instance	141

Multiple Session Manager instances sharing a configuration	142
Debugging and problem diagnosis facilities	147
User and session problems	147
Script and menu problems	147
Locked keyboards	147
Additional commands	147
Further information	148
Access from CICS to Session Manager	149
Chapter 6 Performing a basic configuration	151
Before you begin	152
How configuration is achieved	152
The configuration	153
Customer, samples and system configuration datasets	155
Classic systems	155
OLA systems	155
Control statement summary	159
Basic statements required	162
Supplied statements	162
Suggested updates to your system	164
Chapter 7 Setting up applications	167
Choosing an ACB	168
UNBIND WAIT applications	176
Best fit ACB allocation for RANGE statements	178
Choosing a LOGMODE	184
MTS parameters	186
User-level session ACB support	187
The Virtual Terminal Masking facility	189
Overview	189
Specifying VTM scripts	189
Maintaining VTM entries	189
Implementing the VTM facility	193
VTM with multiple Session Manager instances	193
Chapter 8 Session Manager user exit processing	197
Overview	199
User exit points	200
User exits flowchart	200
Installing the exits	204
Standard exit conventions	205
Return codes	205
Exit load module	205
Register conventions	205
Parameter list	206
Initialization exit point (E01)	208

Configuration statement processing exit point (E05)	209
Application status change exit point (E06)	211
Timer interval exit point (E08)	212
CALLEXIT invocation point (E09)	213
Logon exit point (E11)	215
Signon validation exit point (E21)	217
Signon completion exit point (E22)	221
Input 3270 datastream exit point (E25)	223
IDLEDISC/IDLELOGOFF/IDLELOCK timer expiry exit point (E26)	224
Return codes	224
Signoff exit point (E29)	225
Slave session pre-initiation exit point (E31)	227
Slave session post-initiation exit point (E33)	229
Output 3270 datastream exit point (E35)	231
SIDLTIME timer expiry exit point (E36)	232
Return codes	232
Slave session termination and SMF record exit point (E39)	233
User exit replacement exit point (E71)	241
Session switch exit point (E79)	242
Closedown exit point (E99)	245
Variable access	246
Use of user-defined variables	247
Linkage conventions	247
Input to the routine	247
Output from the routine	249
Producing statistics for NETSPY and ETE	251
Input and output 3270 datastream exit points	252
Overview	252
Characteristics of the input/output exits	252
Storage Obtain/Free routine	258
3270 datastream areas	259
Variable Access routine	259
Issuing Session Manager commands	260
Issuing the HARDCOPY command	261
Setting up an output message	261
Sample exit	261
Sample macros	261
Input 3270 datastream exit point (E25)	272
Output 3270 datastream exit point (E35)	276
The Multiple Exit Driver	279
The Exit Control panel	279
Setup steps	280
How the Multiple Exit Driver works	280
Exit scripts	282
Exit parameters	282

Valid exit SCRIPT parameters	282
Use of ISZCMD	282
Return codes	283
User audit facility	284
Introduction	284
E33 Session post-initiation exit	284
E39 Session termination exit	284
E25 Input 3270 datastream exit	285
Customizing and installing the exits	286
Timeout exits facility	288
Introduction	288
Customizing and installing the exits	288
Chapter 9 Defining security and implementing dynamic menus . 289	
Defining security	290
Designing the signon panel	290
User authentication	290
Internal user authentication – the USER statement	291
External user authentication – VSAM and ESM	291
Sample user exits	292
Exit script ISZE21PH	292
Using VSAM	293
ISZE21VM and ISZE22VM exits	293
Using the Session Manager ADMIN panel	293
Using an External Security Manager (ESM)	293
FAQs	294
Implementing Dynamic Menus	306
How is the Dynamic Menus facility provided?	306
Sample user exit	306
Storage use and performance	306
FAQs	307
Dynamic Menu summary	312
Example configuration statements	312
Chapter 10 PassTickets and Certificate Express Logon 315	
PassTickets	316
Certificate Express Logon	318
Chapter 11 The Shared Userid facility 319	
Overview	320
Operational considerations	321
Summary	323
Chapter 12 Network Data Minimiser (MISER) 325	
MISER for outbound data streams	326
MISER for inbound data streams	326
Determining inbound savings	326

Incompatibilities between MISER and non-miser operation	327
MISER overhead	328
QUERY STATS output for MISER	328
Restrictions to the use of MISER	330
Data stream compression in Session Manager	331
Overview	331
COMPRESS and MISER options	331
Using both COMPRESS and MISER	332
Determining compression savings	332
Data compression and remote sessions	333
Recommended MISER definitions for remote sessions	334
Summary	335
Chapter 13 How to set up and use the CICS front-end	337
Overview	338
CICS definitions	339
Accessing Session Manager using the CICS front-end	342
Querying the CICS front-end	345
Terminating the CICS link	346
Console messages from the CICS agent	347
Modifying the supplied starter transaction program	348
Chapter 14 Session Manager and TCP/IP	351
Introduction to TCP/IP	352
Remote nodes	352
Ports	353
TCP/IP software	354
The TELNET application	354
Support for TCP/IP in Session Manager	355
Overview	355
Starting and stopping TCP/IP	358
Configuring TCP/IP client sessions in Session Manager	358
TELNET client session modes	359
VT220/VT100 emulation	360
Line mode operation (NVT)	366
EBCDIC and ASCII translation (line mode and VTxxx mode)	371
TN3270E operation	371
Chapter 15 Session Manager networking	373
Overview	374
Establishing remote application sessions	375
Reducing network load by using Session Manager networking	375
The REMOTE session parameter	377
Session Manager QUERY command and remote sessions	377
Script and exit execution sequence	379
Selection sequence for session parameter definition	380

Exceptions in session parameter selection sequence	380
Variables and remote sessions	381
Script processing and remote sessions	382
User affinity	383
Specifying remote commands	384
Summary	385
Chapter 16 Parallel Sysplex support	387
Overview of facilities	388
Workload balancing	388
Sharing of the Session Manager configuration	388
Sysplex configuration and operation	388
Setting up for a Sysplex operation	392
Sysplex Networking	392
LINK statements in a Sysplex	393
User reconnection	395
User reconnection migration	395
VTAM application session recovery	398
Overview	398
FAQs	399
Principles and components	410
Recovery levels	412
Application granularity	413
ACB and RANGE statements	413
Recovery planning	414
Example recovery configuration	415
VTAM application session recovery configuration	419
How to configure a Controller	419
How to configure a Standby Controller	419
How to configure a Standby Instance	419
How to configure a Primary Instance	419
How to configure a Primary Instance which is also a Standby Instance	419
How to set the session recovery level	420
SYSTEM statement SESACB considerations	420
ACBs and RANGE statements considerations	420
VTAM MNPS settings	420
How to set the Sysplex audit log	421
Other SYSPLEXGROUP parameters	421
Sample signon validation exit script (E21)	422
Summary	424
Chapter 17 Using Session Manager as a 'front-end'	425
Overview	426
Impact on terminal definition and logon	427
Using a selective signon to Session Manager	428
Logmode table entry name implications	430

Summary	431
Chapter 18 The Application Builder feature	433
Overview	434
Application Builder and remote sessions	435
Chapter 19 Monitoring performance statistics	437
Performance Monitor	438
The Performance Statistics panel	438
Implementing the Performance Monitoring routine	439
Response Time Monitor	440
Examples of response time statistics	441
Notes on response time monitoring	442
Extracting response time statistics	443
Chapter 20 Converting a Classic to an OLA system	445
Overview	446
Datasets used by Session Manager	446
Summary of conversion procedure	446
Conversion procedure	448
How reconfiguration is achieved	448
Configuration DDNAMEs	448
The OLA Enabler	451
Installing an OLA system	451
Running the OLA Enabler	451
Definitions created by the OLA Enabler	452
Assignment of session types to sessions	453
How to load a Language Pack	453
OLA Enabler parameters	453
Setting up OLA for single and multiples Session Manager instances	460
Creating the Session Manager started task	460
Appendix A Structure connection specifications	461
Index	463
Bibliography	473
IBM Session Manager library	473
Accessibility	475
Accessibility for people with disabilities	475
Changing font, color and display settings	475
Using Session Manager with a screen reader	475
Documentation	475
Notices	477
Trademarks	478

Sending your comments to IBM479

About this manual

This is the *Installation and Customization* manual for IBM® Session Manager for z/OS®. You should read it if you are responsible for installing, configuring, or supporting the product. The manual is based on the *Installation and Getting Started* manual but significantly increased in size by the inclusion of material from the *Facilities Reference* manual. The *Facilities Reference* manual has now been renamed as the *User and Administrator* manual.

Note Any references in this manual to “Session Manager version 1.3.15” and to “1.3 Functional Enhancement PTF 3” are synonymous.

Installing Session Manager

Session Manager can be executed in two modes:

- *Classic* – you will need to edit the configuration statements through ISPF or a similar editor

or

- *Online and Batch Administration (OLA)* – configuration is performed through the security-controlled user-friendly OLA interface.

The ‘Installation’ chapters give the operating system prerequisites for successful installation, along with instructions for performing a new first-time (basic) installation in either Classic or OLA mode, converting to an OLA system, applying a Functional Enhancement PTF and upgrading to a new version or release.

Configuring Session Manager

The ‘Post-installation configuration issues’ and ‘Performing a basic configuration’ chapters provide advice and assistance on configuring the product. Session Manager can be tailored to Installation requirements using:

- **Product configuration statements**

In general, the extreme flexibility of the configuration statements is found to cater for most user requirements. Full descriptions of each Session Manager configuration statement and each Session Manager command can be found in the *Technical Reference* manual.

- **Online and Batch Administration**

Online Administration: Instead of supplying product configuration statements directly, Online Administration (hereafter “OLA”) enables administrators and end-users of Session Manager to tailor the product using a series of menus, lists and attribute display panels.

Batch Administration: If many changes are required to a large number of configuration definitions, this capability enables administrators and end-users of Session Manager to tailor the product using a batch job.

For more information, see the *Online and Batch Administration* manual.

- **Panels and Scripts**

Installation-specific facilities can be created using Session Manager Panels and Scripts, and conditional logic can be incorporated using the product’s Panel and Script Language (TPSL). For details, see the *Panels, Scripts and Variables* manual.

To meet particular needs, a User exit is available, containing several exit points and access to certain variables, enabling user code to be executed. For details, see ‘Session Manager user exit processing’ on page 197.

For more information, see ‘How configuration is achieved’ on page 152.

External Security Managers

External Security Managers (hereafter ‘ESMs’), such as RACF[®], can be used with Session Manager to authenticate users, set their authorization level and OLA security class, and determine which applications a user can access.

For details, see the ‘Defining security and implementing dynamic menus’ chapter in the *Installation and Customization* manual.

Session Manager documentation

The following documentation accompanies Session Manager:

Manual	Purpose
<i>Installation and Customization</i>	Goes through the steps required to install the Session Manager software, and provides general information on the methods and options available to configure and operate your system.
<i>User and Administrator</i>	Describes in detail the features and facilities provided by Session Manager.
<i>Online and Batch Administration</i>	Explains the set-up and configuration of OLA, how to use the interface, and how to utilize both OLA and Batch Administration to modify the Session Manager configuration.
<i>Technical Reference</i>	Provides a detailed reference for Session Manager commands and configuration statements, along with problem diagnosis assistance.
<i>Quick Reference</i>	Provides a quick way to find the correct syntax for commands, configuration statements, and variables, without detailed explanations.
<i>Panels, Scripts and Variables</i>	Gives a detailed technical account of defining panels, using scripts and variables, and the product's Panel and Script Language (TPSL).
<i>Messages and Codes</i>	Contains explanations of all messages issued by Session Manager, and the actions that should be taken.

Additionally, the *Program Directory* contains information for systems programmers about the program material and procedures for installing Session Manager under z/OS.

New users should review the *User and Administrator* manual to gain an understanding of Session Manager concepts, and Technical Programmers should review the *Installation and Customization* manual in order to tailor the product to the Installation's requirements.

In general, the extreme flexibility of the configuration statements is found to cater for most user requirements. Panel and script definitions may be provided with conditional logic using the Session Manager Panel and Script Language and numerous variables are provided to view and modify a wide variety of data. If any particular needs cannot be met, a user exit is available, containing several exit points and access to certain variables, enabling user code to be executed. Full details of the User Exit are supplied in the *Installation and Customization* manual.

Conventions

The following typographic conventions are used:

boldface	Indicates a command or keyword that you should type, exactly as shown. When mixed case is used, the element in upper case represents the shortest acceptable form. For example, <code>MSGsuffix</code> can be abbreviated as far as <code>MSG</code> .
<i>italics</i>	Indicates a variable for which you should substitute an appropriate value.
monotype	Indicates literal input and output.
Ctrl+D	Indicates two or more keys pressed simultaneously.
[]	Brackets surround an optional value.
	Vertical bars separate alternative values from which you must make a selection.
...	Ellipsis indicates that the preceding element may be repeated.
@	Some commands or key sequences make use of the 0x7C (that is, x'7C') character. When using the English language code page, this character is displayed as the @ sign, but may be displayed as a different character in some other code pages. In this document, the 0x7C character is always presented as the @ sign. You should enter the appropriate 0x7C character symbol for the code page you are using.

New features

See:

'New in Version 3.100 (Session Manager Release)' on page 19

'New in Version 2.2.05 (IBM PTF: UK55397, APAR: PM05796)' below

'New in Version 2.2.00 (Session Manager Release)' on page 31

'New in Version 2.1.05 (IBM PTF: UK37522, APAR: PK65596)' on page 36

'New in Version 2.1.00 (Session Manager Release)' on page 36

'New in Version 1.3.15 (IBM PTF: UK20403, APAR: PK33701)' on page 39

'New in Version 1.3.10 (IBM PTF: UK15956, APAR: PK21439)' on page 43

'New in Version 1.3.05 (IBM PTF: UK11656, APAR: PK16122)' on page 45

'New in Version 1.3.00 (Session Manager Release)' on page 48

New in Version 3.100 (Session Manager Release)

New PC interface An Eclipse-based interface is provided with this release. This has the same capabilities as the existing System Management Menu facility, the Help Desk facility and the Sysplex[®] Summary and Menu facility but is task and object orientated in terms of usability. A small number of the 3270-based functions are not supported in this release.

A LOCALNODE value must be specified on the SYSTEM statement. LOCALNODE has no value by default. Also, a USER statement must exist for each Eclipse user. Users must sign on through the signon dialogue box even if SIGNON NO is specified on the SYSTEM statement.

Support for External Security Manager password phrases Support is now provided for password phrases (passphrases) to be used, in place of passwords, when authenticating users with an External Security Manager.

Change passphrase facility The change passphrase facility can be added as a selectable session. Sample panels, Profile, Session and APPL are supplied. This facility should only be used on Session Manager instances that are configured to use an External Security Manager, such as RACF. See the *User and Administrator* manual for further details.

New/changed exit samples A new E21 sample script ISZE21PH is provided. This supports the use of External Security Manager password phrases, enables the Session Manager signon process to be multi-threaded and allows multiple PROFILES (to a maximum of eighteen) to be assigned to a user. The existing E21 exit cannot be run together with this new exit. The new exit supports all the current functions of the existing assembler E21 exit.

Submenus

It is now possible to configure multiple levels of Session Manager menus. See changes to CMD parameter and session variable `s_sm_prof`. Also see the *User and Administrator* guide for further details.

Improved installation process

The JCL used in the installation process and product startup has been improved to remove the risk of customer-modified configuration members being overwritten by supplied members with the same name.

Revised startup JCL

Sample members ISZCOLA and ISZCSTRT no longer have DD statements for DATAFILE, SITEINFO and ADDRINFO.

Mutually exclusive parameters in OLA

When Session Manager is checking for the mutually exclusive parameters SHARE, SHARESESS and SHAREDISC, if a user attempts to turn one of them on the OLA now checks the settings in effect at the current level, taking account of inheritance and pending changes of the other parameters. If either is set ON or YES, the validation fails and no processing will occur until a valid value is entered. An error message is issued to indicate the reason.

Deleting/resetting subparameters in OLA

Any attempt to delete or reset a major subparameter is rejected with message ISZ3017E. Any attempt to select a minor subparameter when its major subparameter is not coded (and does not have a new value pending) is rejected with message ISZ3115W.

OLA attribute panels

The OLA attribute panels now state when the change to the attribute becomes effective, once it has been activated.

OLA 'List of LOCAL users' controlled by the ESM

This provides the ability to restrict the 'List of LOCAL users' displayed to userids which reside in the OLA user's default External Security Manager user group.

DEFPROFILE changes

Up to eighteen profiles can now be assigned/configured as the default.

New OPTION parameter

This new parameter can be used in the OPTION statement.

Parameter	Description
MDPROF	New parameter. If MDPROF M is specified, each DEFPROFILE defined on the SYSTEM statement will be added as a default PROFILE. Up to eighteen DEFPROFILES may be specified. If MDPROF L is specified and multiple DEFPROFILES defined, then only the last DEFPROFILE specified will be used as a default PROFILE. If MDPROF is not specified then MDPROF L is assumed.

New/changed SYSTEM parameters

These additions/changes have been made to parameters on the `SYSTEM` statement:

Parameter	Description
DEFPROFILE	Up to eighteen <code>DEFPROFILES</code> when the <code>OPTION</code> parameter <code>MDPROF</code> is set to <code>M</code> .
DIAGS	New subparameter of <code>ECLIPSESERVER</code> (see below). If set to <code>YES</code> , specifies that trace data is produced when the Eclipse feature is being used.
DUMPBASE	New subparameter of <code>DUMPGDG</code> parameter (see below). Mandatory if <code>DUMPGDG</code> set to <code>YES</code> . Specifies the dump GDG base name which forms the prefix of the dump GDG dataset name to be allocated during Session Manager startup.
DUMPGDG	New parameter. If set to <code>YES</code> specifies that dumps should be written to a GDG dataset. Note that dumps written using the <code>DUMP</code> command will also be written to the GDG dataset.
DUMPSMSCLS	New subparameter of <code>DUMPGDG</code> parameter (see above). Specifies the SMS data class if the dump GDG is SMS managed.
DUMPPSPACE	New subparameter of <code>DUMPGDG</code> parameter (see above). Specifies primary allocation, in cylinders, to assign to the dump GDG dataset.
DUMPSSPACE	New subparameter of <code>DUMPGDG</code> parameter (see above). Specifies the secondary space allocation, in cylinders, to assign to the dump GDG dataset.
DUMPUNIT	New subparameter of <code>DUMPGDG</code> parameter (see above). Specifies the type of device upon which the dump GDG dataset will reside.
ECLIPSESERVER	New subparameter of the <code>TCP</code> parameter. Specifies the port on which the Eclipse server listens for connection requests. <code>TCP YES</code> must also be specified.
IMSCONVERTC	New parameter. If set to <code>YES</code> processing is similar to <code>IMSCONVERT</code> , but the processing is conditional on the <code>BIND</code> for the session not specifying a default 3270 screen size of model 2.
OLA_DEFER_USERS	New parameter. Specifies whether all <code>OLA USER</code> statements are to be loaded at Session Manager Start time or only loaded when each user signs on.
PASSPHRASE	New subparameter of the <code>SECURITY</code> parameter. Specifies whether a passphrase can be used instead of a password when authenticating a user with <code>ESM</code> at Session Manager signon.

Parameter	Description
PASSWORDREQ	New subparameter of the new PASSPHRASE parameter (above). Used to indicate if a password must also be provided, along with a passphrase, at Session Manager signon.
RCMDTIMEOUT	New parameter. Specifies the number of minutes after which a command sent to a remote system will timeout.

New parameters

This new parameter can be used in the SYSTEM, USER, TERMINAL, PROFILE and APPL statements.

Parameter	Description
REJBB	New parameter. If set to YES, and a request is received from an application with an SNA Begin Bracket and the session is already in a bracket, then the request is rejected by negative response and sense code.

New/changed common end-user parameters

These additions/changes have been made to common end-user parameters:

Parameter	Description
DAPPLESMAUTH	New parameter. If the E22 exit is active, if set to YES, specifies a call is issued to the External Security Manager to check if user is permitted access to the application specified on the ADDSESS command.
ESMOLAGROUP	New parameter. Used to indicate whether a local USER list in an OLA session should be restricted to userids held within the Session Manager user's default External Security Manager group.
ERTIMEOUT	New parameter. Specifies the number of minutes after which an Eclipse request that has not responded is deemed to have timed out.
EUTIMEOUT	New parameter. Specifies the number of minutes of inactivity after which an Eclipse user is logged off.
MAXUSRLOGIN	New subparameter of SHARESESS parameter. Specifies the maximum number of shared sessions permitted

New session options

These additions/changes have been made:

Parameter	Description
ISZSM	New subparameter of CMD parameter. Enables you to indicate that the session is a sub-menu placeholder by specifying CMD 'ISZSM <i>profilename</i> ' where the <i>profilename</i> specified must be an existing PROFILE.

Parameter	Description
STOPINH	New subparameter. Used to stop a common session parameter setting from being inherited in its entirety from a higher level of definition, for example, PROFILE setting. Value will revert to the default.
STOP_INHERIT	New subparameter of ADDSID and SAUTOSEQ statements to suppress a specified value if several have been defined in a USER, PROFILE or session definition.

New/changed commands

These command parameters have been added or changed:

Parameter	Description
NONQUAL	New parameter of QUERY command. If specified, must precede <i>user_pattern</i> which must be the only other parameter specified. Only details of non-qualified users matching the pattern are returned.
UPDATE GCMVS	New command that allows user to dynamically update the z/OS system symbols designated by variable names <i>GC_MVS_symbolname</i> .
REFRESH	New command used on OLA panel that is used to display member lists for all OLA PDSEs. Command causes OLA to reread the PDSE and redisplay the member list from the beginning again, taking into account any changes that have taken place since the list was originally built.

New/changed SCRIPT functions/parameters

These functions have been added:

Function	Description
SPLXLOG	New function on LET statement. Specifies the block id of the record in character format.
APPLINF	New function on LET statement. Returns information about VTAM applications known to Session Manager.
BCASTINF	New function on LET statement. Returns information about held broadcasts.
BLKICNV	New function on LET statement. Converts a 16-byte character block id to the SPLXLOG 8-byte hex value.
BLKOCNV	New function on LET statement. Converts the SPLXLOG 8-byte hex block id to a 16-byte character value.
CURRYEAR	New function on LET statement. Returns the current year, YYYY.
DLOGINF	New function on LET statement. Returns entries from the internal DLOG message table.

Function	Description
ESMAUTHOLA	New function that returns the AUTH level and OLACCLASS assigned to the username by the ESM. On completion always sets <code>t_rc</code> to 0 and sets a reason code in <code>t_result</code> .
ESMAUTHUSER	New function that authenticates the current USER and implements a new password or passphrase with the ESM. On completion sets a return code in <code>t_rc</code> .
ESMCHKUSERGROUP	New function checks if the provided <i>username</i> resides in the ESM group of the current user. On completion sets <code>t_rc</code> to a return code, where: 0 User resides within group. 4 User does not reside within group.
ESMGROUP()	New function that interrogates the ESM to build an internal list of all userids which reside in the current users ESM group. On completion sets <code>t_rc</code> to a return code, where <code>t_rc</code> can be: 0 Successful completion. 4 List not built.
ESMSIGNON	New function that performs the ESM checks required for user signon to Session Manager, based on the configuration settings of the SECURITY parameter and subparameters. After execution, sets a return code in the variable <code>t_rc</code> and a reason code in the variable <code>t_result</code> .
EMSVERIFYUSER	New function that verifies the current user with the supplied password or passphrase. On completion sets <code>t_rc</code> to a return code.
GFSACTIV	New function on LET statement. Returns current GFS state.
GFSSTATS	New function on LET statement. Returns a line of output from the GFS STATS processor.
GFSSTOR	New function on LET statement. Returns a line of output from the GFS STOR processor.
GFSUSAGE	New function on LET statement. Returns line of output from the GFS USAGE processor.
GROUPINF	New function on LET statement. Returns information about defined Session Manager groups.
LOCKINF	New function on LET statement. Returns information regarding the 'lock' status of a specific terminal.
MDTHON	New parameter to honour MDT flags set by an application in a non-MISER system for duration of the script.
SESSINF	New function that returns information regarding a specific session.
SHPROF	New function which returns for the specified sub-menu PROFILE either the sub-menu PROFILE name of the higher sub-menu or blank if there is no higher sub-menu.

Function	Description
SMDESC	New function that returns the description of specified sub-menus PROFILE.
SMSTATUS	New function that returns 0 if a session in the specified sub-menu PROFILE or lower sub-menu is active or 4 is no session is active.
USERINF	New function on LET statement. Returns information regarding a specified user.
XMLTRANS	New function on LET statement. Translates hex values in the string to X'4B. (full stop. C:?)

New variables

These global, session, detail and user variables have been added:

Variable	Description
t_e21_script	(Global variable). Set to Y if the ISZE21PH exit script is active, otherwise set to N.
s_imsconvertc	(Session variable). If set to Y processing is similar to s_imsconvert variable, but the processing is conditional on the BIND for session not specifying a default 3270 screen size of model 2.
s_sm_prof	(Session variable). Returns either the sub-menu PROFILE name the session is to be displayed in or blank if session was defined at top level.
s_rejbb	(Session variable). If set to Y and a request is received from an application with an SNA Begin Bracket and the session is already in a bracket, then request is rejected by negative response and sense code.
t_day	(Global variable). Returns the capitalized day of the week.
t_month	(Global variable). Returns the capitalized month of the year.
t_maxusrlogin	(User variable). Specifies maximum number of logins permitted for the user.
t_ola	(Global variable). Returns Y or N to indicate whether the system is running in Online Administration or Classic mode.
t_olagroup	(User variable). Returns the name of the ESM group that the user was connected to during signon.
t_passphrase	(Global variable). Returns the value 'Y' or 'N' reflecting the value of the SYSTEM/SECURITY parameter PASSPHRASE.
t_passwordreq	(Global variable). Returns the value 'Y' or 'N' reflecting the value of the SYSTEM/SECURITY/PASSPHRASE parameter PASSWORDREQ.
t_nphrase	(User variable). Contains any new password phrase entered at Session Manager signon.

Variable	Description
t_phrase	(User variable). Contains the users password phrase entered at Session Manager signon.
t_submenus	Returns Y if the user has submenus configured., otherwise set to N.
t_safrc	(User variable). Contains the SAF return code from the user signon.
t_esmrc	(User variable). Contains the ESM return code from the user signon.
t_tss_emsg	Set to 'Y' if the E21 exit requires ESM messages generated during user verification to be returned to the exit.

Messages

A number of messages have been added or amended. See the *Messages* manual for further details.

User exit entry point 79 ACB value

When a Session Manager function is being switched to, the ACB field will contain a value indicating the function type.

New in Version 2.2.05 (IBM PTF: UK55397, APAR: PM05796)**Screen lock function**

The screen can be locked by three methods: a) by using a PF/PA key or keystroke sequence (SAUTOSEQ), or b) automatically, based on timers (IDLELOCK), or c) by command (LOCKTERM). When locked, entry of the correct password is the only action that unlocks the screen.

'User session conceal' facility

A user can prevent one or more (unwanted) sessions from being displayed on their menu when, for example, an External Security Manager such as RACF determines a menu session list containing 'noise' entries that are rarely if ever used.

Improved performance and storage use

A new parameter DROP_SESSION is processed after the E22 exit, to free session storage and thus improve performance and storage use for dynamic menus. The performance of the SESTYPE parameter has also been improved.

New/changed exit samples

Existing exit sample ISZE22DM has been changed to accommodate processing of the new 'drop session' parameters.

CV64 structure propagation

The CV64 structure, containing the client's IP address and port is created by z/OS Communication Server TN3270, can now be passed onto the client's Session Manager applications. Similarly, if the Session Manager TN3270 Server is used a CV64 structure can now be passed onto the client's Session Manager applications. See the new SYSTEM statement parameter CV64 for details.

E26 Timeout exit A new exit point (Exit Script only) invoked when either of the IDLELOGOFF, IDLEDISC or IDLELOCK timers expire. The exit point may decide to extend the timeout period or allow it to expire, and may use OUTSCAN to inspect the current visible application session screen (if available). For additional details, see ‘Timeout exits facility’ on page 288.

E36 Timeout exit A new exit point (Exit Script only) invoked when the SIDLTIME timer expires. The exit point may decide to extend the timeout period or allow it to expire, and may use OUTSCAN to inspect the application session’s screen. For additional details, see ‘Timeout exits facility’ on page 288.

Idle processing on outbound The control of when an idle session/user gets terminated/disconnected now operates for output (as well as input) in order to keep sessions running that are receiving output at regular intervals from an application (for example, a monitor like OMEGAMON).

Allow variable substitution in OLA parameters Variable substitution is now allowed for the GENRESNAME parameter and the length of this parameter has been increased from eight to 50 characters.

Audit log to GDG The Session Manager audit log can now be written to a Generation Data Group (GDG) dataset.

Support for Read Partition Query List When running session scripts, Session Manager provides support for the 3270 Write Structured Field Read Partition Query List command in a transparent manner, that does not require changes to session scripts.

AUTOSELECT on applname AUTOSELECT, which is used to specify a session number to be automatically selected when the user logs on, has been enhanced so that a session can be automatically selected based on the application name.

New install job A new install job, ISZCOJOB, is provided to facilitate OLA upgrades from level 2.2.00 and higher.

Session Manager TN3270 Server Users who access Session Manager through the Session Manager TN3270 Server would normally cause message 249 to be issued when they sign on. However, if the new SYSTEM statement parameter TN3270_MSG4049 is specified then message 4049, containing the TN3270 Client’s IP address and port, will be issued in place of message 249.

New/changed common enduser parameters These additions/changes have been made to common enduser parameters:

Parameter	Description
AUTOSELECT	The AUTOSELECT parameter has been enhanced to enable a session to be automatically selected at logon based on an application name.

Parameter	Description
IDLELOCK	New parameter. Supports automatic screen locking based on timers. Specifies the interval, in minutes, since the last input was received from the terminal, after which the 'lock' internal session is activated.
DIRECTION	New subparameter on the IDLEDISC, IDLELOGOFF and IDLELOCK parameters. Controls whether (activity on) input alone or both input and output datastreams will inhibit a timeout.

New common session parameters

These additions have been made to common session parameters:

Parameter	Description
CONCEAL	Specifies if the user has chosen to conceal the session on their Session Manager menu.
DIRECTION	New subparameter on the SIDLTIME parameter. Controls whether (activity on) input alone or both input and output datastreams will inhibit a timeout.
DROP_SESSION	Specifies whether a routine entered after return from the E22 exit will free session storage. Used to improve storage use and performance for dynamic menus.

New SCRIPT parameter

This parameter has been added to the SCRIPT statement:

Parameter	Description
RPQLSCRIPTAUTO	If set to YES, then, transparently to the script, any Write Structured Field Read Partition Query List command encountered is sent to the terminal, a response solicited, and the response passed on to the application.

New OPTION parameters

These parameters, and corresponding initialization options, have been added to the OPTION statement:

Parameter	Description
E26	Parameter for IDLEDISC/IDLELOGOFF/IDELock timer expiry exit point (E26). With corresponding initialization option.
E36	Parameter for SIDLTIME timer expiry exit point (E26). With corresponding initialization option.

New SYSTEM parameters

These additions have been made to parameters on the SYSTEM statement:

Parameter	Description
AUDITOGDG	New parameter. If set to YES then specifies that audit log records additionally should be written to an audit GDG dataset.
ATGBASE	Subparameter of the new AUDITOGDG parameter (above). Specifies the audit GDG base name which forms the prefix of the audit GDG dataset name to be allocated during Session Manager start up or on issuing the SPIN AUDITGDG command.
ATGUNIT	Subparameter of the new AUDITOGDG parameter (above). Specifies the type of device upon which the audit GDG dataset will reside.
ATGSMSCLAS	Subparameter of the new AUDITOGDG parameter (above). Specifies the SMS data class if the audit GDG is SMS managed.
ATGSPACE	Subparameter of the new AUDITOGDG parameter (above). Specifies the primary space allocation, in cylinders, to assign to the audit GDG dataset.
CV64	New parameter. If specified then TN3270 users who access Session Manager directly through its TN3270 Server will also provide a CV64 structure to their VTAM applications.
DYNMDROPSESSION	New subparameter of the SECURITY parameter. Used by the E22 exit to determine whether sessions to which a user does not have access are dropped or hidden.

New/changed commands

These additions/changes have been made to commands:

Command	Description
CONCEAL	General user command. Used to dynamically set visibility of session on menu.
REVEAL	General user command. Used to dynamically set visibility of session on menu.
LOCKTERM	Enables users to lock their terminal, which can then only be unlocked by successfully entering their password.
SPIN	The SPIN command has a new flavour, SPIN AUDITGDG, to close current GDG dataset and allocate, catalogue and open the next (+1) generation without restarting Session Manager.

New command parameters

These command parameters have been added:

Parameter	Description
E26	New parameter on UPDATE EXIT administrator and privileged user commands.

Parameter	Description
E36	New parameter on UPDATE EXIT administrator and privileged user commands.

New variables

These global, session detail and user variables have been added:

Variable	Description
s_conceal	(Session detail variable). Part of a 'user conceal' facility, whereby a user can prevent one or more (unwanted) sessions from being displayed on their menu. Indicates whether or not a session is visible.
s_dropssess	(Session detail variable). 'Y' or 'N' according to the value of the common session parameter DROP_SESSION.
t_dynmdropssess	(Global variable). Used by E22 exit. Contains the drop attribute value 'Y' or 'N'.
t_user_qual	(User variable). A 'user qualifier' created from the last four characters of the terminal LU name, or is an eight-digit number in the range 1-99999999.

New user exit variables

These user exit variables have been added:

Variable	Description
ec26_ttype	Exit point E26 variable. Terminal task type.
ec26_term	Exit point E26 variable. Terminal name.
ec26_logm	Exit point E26 variable. Logmode name.
ec26_user	Exit point E26 variable. Userid.
ec26_prof	Exit point E26 variable. Profile name.
en26_auth	Exit point E26 variable. Authority level.
ec36_ttype	Exit point E36 variable. Terminal task type.
ec36_term	Exit point E36 variable. Terminal name.
ec36_logm	Exit point E36 variable. Logmode name.
ec36_user	Exit point E36 variable. Userid.
ec36_prof	Exit point E36 variable. Profile name.
en36_applid	Exit point E36 variable. Application name.

New in Version 2.2.00 (Session Manager Release)

Installation and product reorganization	<p>In previous releases Session Manager was shipped in <i>Classic</i> mode and customers wishing to use Session Manager in <i>OLA</i> mode had to install the product in Classic mode, using SMP/E, then run several jobs to convert this installed product to OLA mode, then upgrade their OLA systems. Any maintenance was also supplied only in Classic mode and again further jobs were required to apply this maintenance to OLA systems. However, all these additional jobs were not under the control of the SMP/E environment.</p> <p>To simplify the installation and maintenance process for customers running Session Manager in OLA mode, the product is now <i>supplied</i> in both Classic and OLA modes. To simplify the Session Manager 2.2.00 installation process for OLA mode, and to allow it to be under the control of the SMP/E environment, several major changes to the product have been required.</p> <p>See ‘Product changes – Session Manager 2.2.00 and higher’ on page 58 for details.</p>
Dynamically add session	<p>A session can now be added to, and subsequently deleted from the user’s Session Manager menu. The added session will exist only for the duration of the current Session Manager session.</p>
Native logon	<p>Users can now log on natively to an application from the main menu, by issuing the NLOG command, bypassing any Session Manager activity.</p>
Profiles determined by ESM	<p>The system can now be optionally configured, through the SECURITY parameter on the SYSTEM statement, so that at Session Manager logon the assignment of a user’s PROFILE is based on resource access rules held by the External Security Manager (ESM).</p>
PTKTVAL usage for second userid	<p>An alternative userid, specified by the APPL statement parameter PSTKUser, may now be used when generating a PassTicket for certain applications, and is managed by the External Resource Manager.</p>
PassTicket usage within session request time	<p>An alternative value that is not a VTAM applid, and is contained in APPL statement parameter PSTKAPPL, may now be used during the authentication of PassTickets.</p>
Express logon if Session Manager set up as generic resource	<p>A generic resource name can now be specified for use in verifying the user to the External Security Manager (ESM) in situations where Session Manager is set up as a generic resource and a single signon is being used for the Express Logon Facility (ELF).</p>
OLA commands accepted from console	<p>A valid OLA security class, with the relevant authority, can now be assigned to the console operator, thus allowing OLA commands to be issued from the operator console.</p>

Honour OLA update if END entered	In OLA if the user presses END, any changes that were made will be saved. Previously they were lost unless ENTER was pressed first.
Improvements to OLA RESET and DELETE	The RESET command/selection in OLA will now reset any changes back to their original settings regardless of whether it was a Modify, Add or Delete. Any modifications to a parameter that does not currently exist in the configuration can now be deleted or reset. Previously only a reset was allowed.
Allow variable substitution in OLA parameters	Variable substitution is now allowed for the LOCALNODE, GENERICACB, SHAREAPPL, SIGNONPANEL, SYSPLEXGROUP and ACBRANGE parameters and the length of these parameters has been increased from eight to 50 characters.
Filter for query on System Management menu	The output from a QUERY ALL carried out through the System Management Menu can now be filtered by user so as to reduce the length of the list.
Duplexing of Sysplex log	Data written to the Sysplex log coupling facility can now be additionally written to a staging dataset to ensure that data records are not lost in the event of a Coupling Facility (CF) failure. This is set up by enabling duplexing when defining the logstream and CF structure.
Default APPL definition	Users can specify a default APPL statement which will be used to supply an ACB range to applications known to VTAM but which do not have a specific APPL statement within the Session Manager configuration.
Remote port number to identify user	When using the Session Manager TELNET server (TCP=Yes) the remote port number can be used to uniquely identify a user when multiple users are sharing a session (as opposed to use of terminal name when connecting through the IBM server).
Blank screen after reconnect	Rather than displaying a blank screen in the situation where an application session is being made the visible session and MISER is not in effect, so Session Manager is unable to determine what should be written to the screen, a session script can now be invoked to display a panel inviting the user to press the application 'refresh' key.
HELP ABOUT panel	A HELP ABOUT panel has been added to display the product name and version number.
User audit	A facility is provided to audit (through SMF records) a Session Manager user's activity for a specific application/s.
XYZSCREEN	Provides the same functionality as READSESS but for sessions that do not use MISER.

BRANCH and LAB functions BRANCH and LAB (label) parameters have been added to the product for use in both scripts and panels. The BRANCH instruction allows for switching the processing in a script/panel to another location rather than the next sequential instruction.

New common enduser parameters

These common enduser parameters have been added:

Parameter	Description
DAPPLCHECK	Indicates whether a check is to be made to ensure that the <i>applname</i> entered with the ADDSESS command exists within the Session Manager configuration.
DSESSRANGE	Used to specify a range of session numbers to assign to dynamically added sessions.

New common session parameters

These common session parameters have been added:

Parameter	Description
BLANKSCRIPT	Identifies the session script to be invoked in a situation where an application session is being made the visible session, MISER is not in effect, but Session Manager is unable to determine what should be written to the screen.
PSTKAPPL	Specifies the value to use when generating a PassTicket for the application (some applications do not use their VTAM applid during authentication of PassTickets but use a different value).
PSTKUSER	Specifies the value to use in place of the original userid when generating a PassTicket for the application.

New SYSTEM parameters

These parameters have been added to the SYSTEM statement:

Parameter	Description
DEFAPPL	Specifies a default APPL statement to be used to supply an ACB range to applications that are known to VTAM but do not have a specific APPL statement within the Session Manager configuration.
GENRESNAME	Contains the generic resource name to be used, instead of the Session Manager instance name, when verifying the user to the External Security Manager (ESM).
OPEROLAClass	Specifies the OLA class to be attributed to the console operator.

Changed SYSTEM parameter This parameter of the SYSTEM statement has been changed:

Parameter	Description
SYSPLEXTYPE	<p>This parameter has a new value and a new default value. The new value is N, which is the new default (previously was I). This means there are now two types of Sysplex instance, N and I.</p> <p>An instance with a SYSPLEXTYPE of N allows conventional VTAM sessions RECOVERYLEVEL NONE (and TELNET client sessions and so on) to be used by its users.</p> <p>An instance with a SYSPLEXTYPE of I is the same as N but in addition its users may use RECOVERYLEVEL HIGH and RECOVERYLEVEL INTERMEDIATE type sessions.</p> <p>Note The SYSPLEXTYPE and RECOVERYLEVEL parameters are only relevant to systems that are utilizing the High Level Availability (HLA) facilities within Session Manager.</p>

New SYSTEM subparameters

These subparameters have been added to the SYSTEM statement:

Subparameter	Description
PORTNUMBER	Parameter of the MULTUSER parameter. Specifies that the remote port number should be used during sign-on to uniquely identify the user when the Session Manager TELNET server is in use.
ESMPRFCLNM	Parameter of the SECURITY parameter. Indicates whether profiles should be assigned to a user based on resource access rules held in the ESM.
ESMPRFRSNM	Parameter of the SECURITY parameter. Sets the name of the ESM resource, which will be queried to determine if the user has authority to use a given PROFILE.
ESMPRFACC	Parameter of the SECURITY parameter. Determines whether the user is granted access which allows them to use a given Session Manager PROFILE, if the ESM cannot determine if a user would have access to the resource.

New PROFILE parameters

These parameters have been added to the PROFILE statement:

Parameter	Description
ESMLEVEL	Controls the order in which access to ESM-managed profiles is checked for a user.

New commands

These commands have been added:

Command	Description
ADDSESS	Enables users to add sessions to their Session Manager menu.

Command	Description
DELSESS	Enables users to delete sessions from their Session Manager menu.
NLOG	Allows a user to log on natively to an application from the main menu, bypassing any Session Manager activity, including any scripts.

New variables

These variables have been added:

Variable	Description
t_dapplcheck	(Global variable). Used when adding a dynamic session and if set to 'Y' will check to ensure that an APPL definition exists for the application.
t_dsrng_from	(User variable). Contains a numeric value derived from the 'from' value defined in the DSESSRANGE parameter. Used when adding dynamic sessions.
t_dsrng_to	(User variable). Contains a numeric value derived from the 'to' value defined in the DSESSRANGE parameter. Used when adding dynamic sessions.
t_esmprfclnm	(Global variable). Contains the ESM class name which will be used when checking if the user has access to the Session Manager PROFILE.
t_esmprfrsnm	(Global variable). Contains the ESM PROFILE resource name, which will be prefixed to the Session Manager PROFILE name to provide the resource name to be checked by the ESM.
t_esmprfacc	(Global variable). Determines the user's Session Manager PROFILEs. Used when the ESM cannot determine if a user would have access to the generated resource name.
t_genresname	(Global variable). Determines whether the Session Manager generic resource name is to be used with the Express Logon Facility instead of the Session Manager instance name.
s_blnkscrip	(Session detail variable). Contains a Script name, as defined by the BLANKSCRIPT parameter.
s_pstkappl	(Session detail variable). If set, is used by the PassTicket exit to generate a PassTicket for an application.
s_pstkuser	(Session detail variable). If set, is used by the PassTicket exit to generate a PassTicket for an application.

Interlink support withdrawn Support for Interlink SNS/TCPaccess has been withdrawn.

New in Version 2.1.05 (IBM PTF: UK37522, APAR: PK65596)

Documentation changes

The *Installation and Getting Started* manual becomes the *Installation and Customization* manual (its order number, GC34-6783, is unchanged) and has been roughly doubled in size by receiving material from the *Facilities Reference* manual.

The *Facilities Reference* manual (SC34-6785) becomes the *User and Administrator* manual (with a new order number, SC34-6978). Its content has been significantly reduced by the transfer of material to the *Installation and Customization* manual.

The ‘Setting up applications’ section has been moved from the *Technical Reference* to the *Installation and Customization* manual.

IPv6 support

This version of Session Manager provides support for IPv6. The support is dual stack, that is, accommodating both IPv4 and IPv6. Sites with IPv6 addresses have a fallback IPv4 address for clients/servers that are unable to cope with IPv6.

New in Version 2.1.00 (Session Manager Release)

Note For further details of functions see the *User and Administrator* manual (was *Facilities Reference*). For further details of commands and parameters see the *Technical Reference* manual. For further details of variables see the *Panels, Scripts and Variables* manual.

Sysplex VTAM session recovery

A VTAM application session recovery mechanism has been introduced for use in a Sysplex. It provides for planned Session Manager to Session Manager switches and for session recovery in the event of a catastrophic failure and comprises new configuration parameters and user panels. It is based on the concept of a “hot standby”, using the following Session Manager components: Instance, Standby, Controller and Standby Controller. Individual sessions are defined with a Recovery Level of High, Intermediate or None.

SPLXNODES function

The SPLXNODES function has been enhanced to generate variables describing the status of Sysplex elements that control VTAM application session recovery.

T_RECOVER_DET function

A new TPSL function, T_RECOVER_DET, returns a group of variables containing various counts relating to the recovery level state of USERS, ACBs and Sessions.

New SYSTEM parameters

These parameters have been added to the SYSTEM statement:

Parameter	Description
SYSPLXType	The type of Sysplex instance.
STANDBY	The Standby target.
VERBOSE	Controls the number of messages issued during a SWITCHPLX process or during VTAM application session recovery.

New SYSPLEXGROUP parameters

These subparameters have been added to the SYSPLEXGROUP parameter on the SYSTEM statement:

Parameter	Description
WAITFORCNTLTIME	The time, in seconds, that the Sysplex instance will wait before attempting to communicate with the Controller.
STANDBYTAKETIME	The time, in seconds, that the Session Manager Standby Controller will wait before assuming the role of the Controller.

New APPL parameters

This parameter has been added to the APPL statement:

Parameter	Description
RECOVERYLEVEL	Signifies the recovery level of an application. This can also be specified on the USER/PROFILE/SYSTEM statement.

New SEND command authority setting

The default AUTH for the SEND command has been downgraded to level 1 from level 5. This will allow general users with an AUTH 1 setting to issue basic commands (for example, DEMO) across the Sysplex network through the Sysplex Summary and Menu facility.

Help desk facility

A new Help Desk facility gives Help Desk operators quick access to information about users, allows users and/or their sessions to be cancelled, allows messages to be sent and allows the Spy facility to be invoked.

Change password facility

Users can change their password in the external security manager, such as RACF, through a session from the main menu.

New password-related variables

These variables have been added:

Variable	Description
t_mxcpass	(Global variable). Indicates whether or not the external security manager, such as RACF, supports mixed case passwords.

ESM security check for ACB and terminal names

The supplied E21 exit has been enhanced so that after verifying the userid it will then check against the external security manager that the user has read access authority to the Session Manager ACB and the terminal name.

New SECURITY parameters

These subparameters have been added to the SECURITY parameter on the SYSTEM statement:

Parameter	Description
SIGNONCLASS	Sets the name of the ESM class in which generated resources (a combination of the setting in SIGNONRESNAME and the Session Manager ACB name) will be queried to determine if the user is allowed to logon to this application.
SIGNONRESNAME	Along with SIGNONCLASS, sets the name of the ESM resource, which will be queried to determine if the user has authority to access this Session Manager.
SIGNONACCESS	Determines whether the user is granted access which allows them to sign on to the Session Manager application, if the ESM cannot determine whether a user would have access to the generated resource name (a combination of the setting in SIGNONRESNAME and the Session Manager ACB name).
TERMINALCLASS	Sets the name of the ESM class in which generated resources (a combination of the setting in TERMINALRESNAME and the terminal name) will be queried to determine if the user is allowed to logon to an application from this terminal.
TERMINALRESNAME	Along with TERMINALCLASS, sets the name of the ESM resource, which will be queried to determine if the user has authority to access Session Manager from this terminal.
TERMINALACCESS	Determines whether the user is granted access which allows them to sign on to the Session Manager application from this terminal, if the ESM cannot determine if a user would have access to the generated resource name (a combination of the setting in TERMINALRESNAME and the terminal name).

New security-related variables

These variables have been added:

Variable	Description
t_authclass	(Global variable). Used by the E21 exit. Contains the ESM class name which will be used to determine a user's AUTH and OLACCLASS settings.
t_authresn	(Global variable). Contains the resource name which when appended with the AUTH value will be used to determine a user's AUTH setting.
t_olaresn	(Global variable). Used by the E21 exit. Contains the resource name which when appended with the OLACCLASS value will be used to determine a user's OLACCLASS setting.
t_dynmclass	(Global variable). Used by the E22 exit. Contains the ESM class name which will be used to determine which applications the user has authority to access.

Variable	Description
t_dynmresnm	(Global variable). Used by the E22 exit. Contains the resource name, which will be appended with either the APPL name or APPLID name, will be used to determine which applications the user has authority to access when dynamic menus are being used.
t_signonclass	(Global variable). Used by the E21 exit. Contains the ESM class name which will be used when checking if the user has access to the Session Manager application name.
t_signonresname	(Global variable). Used by the E21 exit. Contains the sign-on resource name, which will be appended with the Session Manager ACB name.
t_signonaccess	(Global variable). Used by the E21 exit. Determines whether a user is allowed to log on to the Session Manager application, if the ESM cannot determine if a user would have access to the generated resource name (a combination of the setting in SIGNONRESNAME and the Session Manager ACB name).
t_terminalclass	(Global variable). Used by the E21 exit. Contains the ESM class name which will be used when checking if the terminal being used by the user has access to Session Manager.
t_terminalresname	(Global variable). Used by the E21 exit. Contains the sign-on resource name, which will be appended with the terminal name.
t_terminalaccess	(Global variable). Used by the E21 exit. Determines whether a user is allowed to log on to the Session Manager application, if the ESM cannot determine if a user would have access to the generated resource name (a combination of the setting in TERMINALRESNAME and the terminal name).

INQINTERVAL processing

The maximum value of the interval (INQINTERVAL) at which Session Manager checks on the status of applications has been increased from 60 to 1440 minutes (one day). A new INQUIRE command, to enable ad hoc inquiries, has been introduced.

New in Version 1.3.15 (IBM PTF: UK20403, APAR: PK33701)

Note Any references in the manuals to “Session Manager version 1.3.15” and to “1.3 Functional Enhancement PTF 3” are synonymous.

Sysplex support

Enhancements to support operation in a parallel Sysplex environment, including workload balancing, configuration sharing, Sysplex networking, automatic user reconnection, Sysplex Group wide BROADCAST and MSG commands and Sysplex Group audit log.

Note To connect pre-1.3.15 Session Manager nodes to 1.3.15 or higher nodes, a PTF must be applied to the pre-1.3.15 nodes. Contact your local representative for details.

New Sysplex user interface

New panels are provided to enable the user to display users and nodes within a Sysplex Group, to enter Sysplex Group commands and to view and navigate the Sysplex Group audit log.

New install job

A new install job, ISZNALOC, is provided to pre-allocate a NODE PDSE with which Sysplex nodes have to register at each start-up (governed by the SYSTEM statement keyword INITIAL_CMD: 'ISZSPNOD').

New SYSTEM statement (SYSTEMMCM)

In an OLA environment a common SYSTEM statement SYSTEMMCM is now utilized. SYSTEMMCM contains SYSTEM parameters that are common across all Session Manager instances.

New SYSTEM parameter

This parameter has been added to the SYSTEM statement:

Parameter	Description
SYSPLEXGROUP	Enables a Session Manager Sysplex group name to be specified.

New Sysplex-related variables

These variables have been added:

Variable	Description
t_global_msg	(Global variable). Controls the scope of the BROADCAST and MSG commands in a Sysplex environment.
t_global_msgdef	(Global variable). Indicates the default scope of the BROADCAST and MSG commands in a Sysplex environment.

New LET statement functions

These functions have been added:

Function	Description
SPLXLOCUSER	TPSL function that enables a script executing in a Sysplex Session Manager system to identify whether a particular user, or SIGNON NO terminal, is active in the Sysplex group.
SPLXLOG	Returns records from the Sysplex global log.
SPLXNODES	Returns information relating to the nodes within a Sysplex group.

VTM and multiple Session Manager instances

Guidance is provided on steps for maintaining the integrity of the VTM PDSE when shared across multiple Session Manager instances, in both a Sysplex and a non-Sysplex environment.

New Enabler parameter

This new optional parameter has been added to the Enabler:

Parameter	Description
NOLINKS	If specified, for when running Session Manager in a parallel Sysplex environment, then no LINK statements will be produced regardless of what is specified on the batlnk, onllnk, comlnk, batname and onlname parameters.

New Enabler-related message

A new message related to the NOLINKS option above has been added.

New and revised Sysplex messages

A number of new Sysplex-related messages have been added, and a few have been modified.

Session autostart

Users can now customize which sessions get auto-started, regardless of whether this is defined on the Session within the configuration and without the user having to take their own copy of the session. This can be achieved by specifying the new SESSAUTOS parameter in a Classic system or by using the new Autostart fields on the Session menus within the Online Administration feature in an OLA system.

New common enduser parameter

A new common enduser parameter has been added:

Parameter	Description
SESSAUTOSAPPL	Enables the user to override the configuration Autostart setting for their sessions based on application name (or command).

New USER parameter

This parameter has been added to the USER statement:

Parameter	Description
SESSAUTOS	Enables the user to override the configuration Autostart setting for their sessions.

New OLA Autostart messages

Some new Autostart-related messages have been added.

New script verbs/parameters

Three new SCRIPT statement Session script verbs (HOME, TAB and ERASE) have been added for use in scripts that communicate with the application (for example, STARTSCRIPT) through INPUT and WAITAPPL for example. They allow the script to navigate around the screen without the need for SBAs on the INPUT statements.

New script/verb related variables

These variables have been added:

Variable	Description
t_recur_script	(Script verb variable) Controls whether TPSL CALL or VCALL to another script can be recursive.
s_status_updates	(Session detail variable) Assigns control to the verb that follows the session script's WAITDATA verb when certain status changes occur.

Bypass OLA member list

If a filter entered on an OLA member list menu exactly matches a member name within the list then that member is selected and displayed without redisplaying a filtered member list.

Filter field on OLA main menus

In OLA, filter fields have been added against required options on the Main Menu, the System Main Menu and the Hardcopy Main Menu. If a filter entered exactly matches a member name then that member is selected and displayed without displaying a filtered member list.

Selection numbers on menus

Options on all menu panels now have selection numbers that can be entered on the command line to select that option (unless the option and its number are excluded by OLA security settings).

User fields/variables in common parameters

Two sets of five user fields (USERDATA1 - USERDATA5 and SESSDATA1 - SESSDATA5) with read-only variables have been added to the Common Enduser Parameters and the Common Session Parameters respectively, for use in scripts, exits and so on. Not used internally by Session Manager.

New user associated variables

These variables have been added:

Variable	Description
t_logdata	Provides the VTAM logon data.
t_signonpanel	Provides name of signon panel.

Ability to change signon panel in E11 exit

It is now possible to change the signon panel name in the E11 exit. As part of this modification the VTAM logon data has also been added to the input parameters to this exit. Please note that there is no checking that the new signon panel actually exists.

To accomplish this a field for the logon data has been added to parameter 4 and a field for the new signon panel name has been added to parameter 6. The logon data field is not modifiable but the signon panel field is. Also new variables have been added to access these fields from scripts and panels. Please see the *User and Administrator* manual (was *Facilities Reference*) and *Panels, Scripts and Variables* manual for further details.

New exit-related variables for E11 These variables have been added:

Variable	Description
ec11_logd	Provides the VTAM logon data within the E11 exit.
ec11_signp	Provides the name of the signon panel within the E11 exit.

Security with E21 exit When displaying a Local User within the Online Administration feature and when using an External Security Manager (ESM) such as RACF with exit E21 active then the AUTH and OLACCLASS values are extracted from the ESM.

PF7/PF8 = BWD/FWD Keys PF7 and PF8 are supplied as BWD and FWD in order to comply with SAA standards.

New/changed configuration samples A new configuration sample, SYSPMENU, has been added. Existing configuration samples, SU, PRODPROF, PRODPRO1, MENU, MENU2 and SIGNONE have been changed.

New/changed exit samples Existing exit sample has been changed.

New/changed JCL samples Two new JCL samples, ISZNALOC and ISZBACOM, have been added. All the other existing JCL samples have been changed.

New terminal description variables These variables have been added:

Variable	Description
t_ipaddr	The TCP/IP address for terminals connected through the IBM CS TN3270 Server.
t_ipport	The TCP/IP port number for terminals connected through the IBM CS TN3270 Server.

New in Version 1.3.10 (IBM PTF: UK15956, APAR: PK21439)

Improved upgrade process Improvements have been made to the upgrade process to ease the handling of the supplied samples. A new job, ISZSCOPY, has been introduced and jobs ISZPALOC and ISZSSEN have been updated. A number of samples have been introduced or changed. Details of these, and guidance on carrying out installs and upgrades for both Classic and OLA systems, are provided in the 'Post-installation configuration issues' chapter in the *Installation and Customization* manual (was 'Post-installation configuration considerations' chapter in *Installation and Getting Started* manual).

MVS 4-byte console IDs Session Manager now supports 4-byte console IDs, introduced with z/OS version 1.4.

- CART support** Session Manager now supports Command and Response Token (CART) when processing messages exchanged with a console.
- Multiple Session Manager instances with OLA** The method for running multiple Session Manager instances with OLA has been changed to employ the technique used by the VTM facility. OLA users need to update their OLAPROF PROFILE and OLA APPL definition to take account of this. VTM users need to add a STARTSCRIPT to their VTM APPL definition.
- OLA 'Confirm Quit'** Previously, if you made changes, OLA forced you to issue:
- ENTER to confirm that you want to QUIT
 - END to cancel QUIT and return
 - SAVE to save changes and QUIT
- even if you subsequently RESET (all of) the changes before returning.
- Two alterations have been made:
- Now OLA is sensitive to whether or not any changes are really outstanding (that is, on an individual basis).
- The flow has been changed so that you automatically exit through SAVE if there are changes outstanding, otherwise you bypass SAVE altogether; in addition, you can CANCEL and discard all unsaved changes at any point.
- OLA RESET option** Previously, to remove a newly added attribute, you RESET it, but once it had been saved, you DELETED it. This caused confusion as to which command to use, so now you remove an attribute with DELETE, regardless of whether it is newly added or has already been saved.
- Additional OLA Help** Additional field-level Help has been added within the OLA system. All the configuration parameters now have online Help available. Help has also been provided for OLA Admin and Userview Session List displays and for OLA Security.
- New common enduser parameter** A new common enduser parameter has been added:
- | Parameter | Description |
|-------------|--|
| SESSPRIAPPL | This parameter enables sessions to be prioritized based on application name (or command). If it is set to Y then OLA will generate a USER statement parameter SESSPRI A (if the session specifies an application) or SESSPRI C (if the session specifies a command). |
| | If neither an application nor a command is specified, or if SESSPRIAPPL N is in effect, then OLA will generate a SESSPRI T (if the session has a SESTYPE associated with it), otherwise it will generate a SESSPRI N. |
- New abbreviation** The COMMANDPRFXVAL user parameter can now be abbreviated to COMMANDP.

New statement This statement has been added:

Statement	Description
DELETE	The DELETE statement is used to delete in-storage definitions from a running Session Manager system. It is designed to operate with OLA and is generated automatically by OLA scripts. The generated DELETE statements are then activated through a PUPDATE command to clear the in-storage references. However, users of Classic systems can create configuration members by explicitly coding their own DELETE statements then performing an UPDATE command.

New variable This variables has been added:

Variable	Description
t_menuntop	(Panel variable) When a scrolling command is issued, t_menuntop is updated to the line number for the new top line displayed on the menu. However, this doesn't happen until the PROCESS section is executed. To obtain this value whilst building the HEADER, CONTENT or TRAILER section, t_menuntop must be used.

Assign SESSPRI by APPL The USER statement SESSPRI parameter now accepts A (application) and C (command) in addition to the existing T (session type) and N (session number).

'?' replaces '#' in ACB substitution Hash (#) is not a recognized key in all countries, so Session Manager now uses the question mark symbol (?) to cause substitution from the user's terminal name. A hash (#) is still allowed in place of a question mark, for backwards compatibility.

Updates to messages Some ISZ-prefixed messages have been amended/added.

New return codes Some new return codes have been introduced (previously there was only a zero return code) to provide additional information about failures.

New in Version 1.3.05 (IBM PTF: UK11656, APAR: PK16122)

Virtual Terminal Masking (VTM) A new panel-driven Virtual Terminal Masking (VTM) facility is provided to allow the ACB name for a selected session to be built using any combination of characters from the userid and session user's terminal name dependent on the APPL name/userid combination.

OLA 'User view' and 'Admin view' In OLA it is now possible for an authorized user see two views of general users' session lists; an 'Admin view' which shows the details/sessions as extracted from the configuration files, and a 'User view' which shows the details/sessions that a user will see on their Session Manager main menu when they log on. These views apply to both 'My USER' and 'Local USER' displays.

- OLA messages** A number of new OLA messages have been added to the *Messages* manual.
- TN3270E client support** Session Manager's TELNET client now allows a URL specifying TN3270E, in addition to the existing TELNET and TN3270 URL types.
- Escape indicator** Escapes are not permitted from some panels (in which the command line prefix is shown as --->). This is to allow for an escape sequence to be entered as *data*. Those panels where escapes *are* allowed have the usual format (===>) for the prefix.
- Updated 'lock terminal' sample** The sample 'lock terminal' facility that is shipped in ISZCSAMP has been updated so that it is compatible with the CMDACTIONKEY parameter (new in this release) and COMMANDPRFXVAL parameter (updated in this release).
- New common user parameter** This common user parameter has been added:

Parameter	Description
CMDACTIONKEY	This parameter allows a Session Manager command action key to be specified on the USER, TERMINAL, LU, PROFILE and SYSTEM statements. When this parameter is specified, SAUTOSEQs that have COMMANDPRX=Y will only be invoked if the SAUTOSEQ sequence is entered and the specified key is pressed. If a user has a CMDACTIONKEY and a COMMANDPRFXVAL in effect then they must wrap their escape sequences and TRANSIDs with these values. They can also optionally wrap Session Manager commands and command scripts with these values.

- New SAUTOSEQ parameter** This parameter has been added to the SAUTOSEQ statement:

Parameter	Description
PARM	This parameter, if set to 'Y' on the SAUTOSEQ statement, causes Session Manager to scan for a parameter of up to eight characters entered after the sequence. If one is found then the variable <code>ucsautoparm</code> will contain the value. The entered parameter is treated as part of the sequence and so removed with it.

- New COMMAND parameter** This parameter has been added to the COMMAND statement:

Parameter	Description
ACTKEY	This parameter, if set to 'Y' on the COMMAND statement, specifies that if the command is invoked anywhere other than in the menu command line, the user's CMDACTIONKEY and COMMANDPRFXVAL must also be specified.

New variables

These variables have been added:

Variable	Description
t_actcmd	(User variable) Enables user to add CMDACTIONKEY to a menu as an input field, allowing user to view and change their command action key.
t_actprf	(User variable) Enables user to add COMMANDPRFXVAL to a menu as an input field, allowing user to view and change their command prefix value.
t_cmd_ok	(Panel variable) Indicates whether or not the user's entered command is left in t_command (command input field) or removed from it.
t_lastsess	(Panel variable) Contains the menu session number of the last session accessed by the user.
s_script_cmds	(Session detail variable) Indicates whether ISZCMD/A supports command scripts, as well as native Session Manager commands. It also indicates that SAUTOSEQ scripts may be run while session scripts are active.
s_tn3270e	(Session variable) Indicates whether or not TN3270E protocols are in use
s_tn3270e_dev	(Session variable) If TN3270E protocols are in use, holds the Server's final device name.
ucsautoparm	(User variable) Contains eight-character variable when PARM set to 'Y' on SAUTOSEQ parameter.

Changed message numbers

message numbers were changed as a result of customer enhancement requests.

pre 1.3.05A (ISZLENMU or ISZCON01)	1.3.05A (ISZLENMS)
ISZ0600	ISZ4012
ISZ0601	ISZ4013
ISZ0602	ISZ4014
ISZ0603	ISZ4015
ISZ0604	ISZ4016
ISZ0605	ISZ4017
ISZ0606	ISZ4018
ISZ0607	ISZ4019
ISZ0608	ISZ4020
ISZ0609	ISZ4021
ISZ0650	ISZ4022
ISZ0652	ISZ4023

pre 1.3.05A (ISZLENMU or ISZCON01)	1.3.05A (ISZLENMS)
ISZ0652	ISZ4024

New in Version 1.3.00 (Session Manager Release)

- CICS® front-end** Changes have been made to the CICS program, profile and transaction definitions specified during set-up of the CICS front-end (documented in the *Installation and Customization* manual (was *Installation and Getting Started*)).
- MASK3 example** A new example is provided in the *Online and Batch Administration* manual to illustrate modification of an existing script or panel, using the MASK3 output mask, during batch administration.
- Global variables** Some missing global variables (t_dynmalog, t_dynmautsthid, t_dynmhide, t_dynmlogmax, t_dynmtype and t_hardenu) have been added in the *Panels, Scripts and Variables* manual.

CHAPTER 1

Deciding on a Classic or an OLA system

This chapter provides help in making the decision to install either a Classic or an Online and Batch Administration (OLA) system.

The subjects covered in this chapter are:

- 'Overview' on page 50
- 'Frequently-asked OLA questions' on page 51
- 'Batch Administration' on page 55
- 'Further information' on page 56

Overview

The configuration of IBM Session Manager for z/OS is controlled using configuration statements. Configuration can either be

- Classic – statements are contained in source members in the PDSs allocated to the DDNAME of CONFIG.
- OLA-enabled – statements are configured and modified using OLA.

If you choose Classic configuration, you will need to edit the configuration statements through ISPF or a similar editor. If you use OLA, some configuration (such as that of panels and scripts) will still need to be edited through ISPF or a similar editor, but most of the other statements must be edited through the security-controlled user-friendly OLA interface. Any Classic configuration that has been converted to an OLA format can not be converted back to a Classic format and, as stated above, must be configured through the OLA interface.

Frequently-asked OLA questions

This section covers many of the most commonly asked questions about using OLA.

What product configuration data can be maintained by OLA?

OLA can be used to maintain all product configuration data, except for:

- Specialized definitions that are stored in the ISZCONxx members (APPLYSU, AUDITROUTE, COPY, INSTALLSU, OPTION, REMOVESU, TRACEROUTE, PATCH, PATCHSU and TRANSTABLE). To update these definitions, you must edit the ISZCONxx members directly.

Definitions stored in the base configuration .SISZCONF dataset, the system configuration .SISZSCNF dataset and the customer configuration .SISZCCNF dataset can *not* be modified or deleted by OLA. To update these definitions, you must edit the members directly after copying the member to your customer library so that your modifications are not lost after future upgrades. Updated members should be saved in library .SISZCUST for Classic and .SISZCCNF for OLA, and a COPY statement for each member added to a new member called for example, USERPANS, and save this member, for OLA users in the .SISZCCNF library and for Classic users in the .SISZCUST library. Then insert a COPY statement for, in our case, USERPANS in each ISZCONxx member. For OLA users, add the COPY USERPANS after the COPY ISZCOMON in the .SISZCCNF library. For Classic users, add the COPY USERPANS at the end of the ISZCONxx member in the .SISZCUST library. (See also the *Online and Batch Administration* manual.)

What changes are required to an existing configuration?

To use OLA when you are using a Classic configuration (that is, all configuration definitions are stored in members of PDS(s) allocated to the DDNAME of CONFIG), you must run the OLA Enabler to implement the new format configuration. For details, see ‘The OLA Enabler’ on page 451.

For new systems, install Session Manager and follow the instructions that set the product up to run in OLA mode.

How is OLA set up, and can it administer multiple instances?

Usually OLA will be used to administer a single Session Manager instance, but it can also be used to administer multiple Session Manager instances. When OLA is administering multiple instances, one dedicated OLA instance modifies a single configuration dataset that is used by all of the Session Manager instances.

Refer to ‘Setting up OLA for single and multiples Session Manager instances’ on page 460 for the appropriate setup instructions.

How is security handled?

The facilities available to a particular user depend on that user’s OLA security class – for details, see the section on security considerations in the *Online and Batch Administration* manual.

What are the main OLA menus, lists and displays?

A summary list of the main OLA facilities appears below; for a detailed description of each these facilities, see the *Online and Batch Administration* manual.

Main Menu

- Display 'My USER' definition menu
 - Display enduser specific parameters
 - Display user's common enduser parameters
 - Display user's common session parameters
 - Display list of sessions associated with user
- Display list of PROFILE definitions
- Display list of local USER definitions
 - Display enduser specific parameters
 - Display user's common enduser parameters
 - Display user's common session parameters
 - Display list of sessions associated with user
- Display list of remote user definitions
- Display list of APPL definitions
- Display list of TERMINAL definitions
- Display list of GROUP definitions
- Display HCOPY definitions menu
 - Display list of hardcopy profiles
 - Display list of hardcopy formats
 - Display a list of hardcopy routes
- Display SYSTEM definitions menu
 - Display list of RANGE definitions
 - Display list of LINK definitions
 - Display list of command authority definitions
 - Display list of SYSTEM definitions
 - Display list of MESSAGE definitions
- Display list of OLA security definitions

Notes

- 1 The facilities available to a particular user depend on that user's OLA security class.
- 2 Hidden application sessions will only be shown in OLA menus, lists and displays if the user's OLA security class allows it.

Is online Help available for OLA?

Yes. To access the online Help, press the PF Key assigned to the Help function. The top level online Help panel will appear:

```

Session Manager      S/Mgr Help - Online Admin Menu      dd/mm/yyyy hh:mm:ss
LU  luname                                     user

This screen gives the main headings for which help is available
for the Online Administration facilities provided in S/MGR

1 GENERAL   - General information
2 SAVE      - Save and Activation information
3 SELECTION - Selection options
4 PFKEYS    - PFKEYS, Commands and Filtering
5 SECURITY   - Security information
6 SESSION   - Session List information

Enter the required number in the command area, or enter the
highlit part of the help heading, to display the appropriate
help information.

***** Press PF11 for next topic *****

PF1:Help PF3:Quit PF4:Return PF6:Top PF7:Bwd PF8:Fwd PF10:Prev PF11:Next

```

From here, specify the OLA area for which you need information and you will be presented with more detailed Help on that topic.

Help on individual parameters is also available by pressing the Help PF Key in panels that display parameter or subparameter details.

What if many changes are required to the configuration?

If many changes are required to a large number of configuration definitions, Session Manager's Batch Administration capability enables you to tailor Session Manager using a batch job. For details, see the *Online and Batch Administration* manual.

Batch Administration also enables you to search the Session Manager configuration files for definitions which match specified search criteria.

For additional information on any of the following subjects, refer to the *Online and Batch Administration* manual.

How are sessions ordered on Session Manager menus?

Application sessions are ordered according to their menu sequence numbers. Users can also accord individual sessions 'priority' so that they are displayed at the top of the Session Manager main menu.

Can specified Session Manager sessions be hidden?

Yes.

Does Session Manager eliminate duplicate sessions?

Yes, Session Manager uses session types to eliminate duplicate application sessions.

What are dynamic Session Manager menus?

For a particular user, Session Manager menus can be configured dynamically from access rules specified using the Installation's ESM.

Is a sample Session Manager menu available?

Sample menus are shipped with Session Manager which:

- Demonstrate the product's menu ordering and sorting ability, and other facilities added in this release. (The sample menus will not display hidden sessions.)
- If your OLA security class allows it, enable you to invoke OLA.

The sample menus can be found in member ISZLENPU, which is supplied in library .SISZCONF.

For all Session Manager menus, including the sample menus, the program code will:

- Prevent you from selecting an application session if its `s_hidden` value is set to Yes.
Note If you create your own menus then you will have to implement some TPSL code to prevent hidden sessions from being displayed. To do this, use the sample menus as a model.
- Order application sessions based on the session's `s_sequence` value.
- Sort application sessions with the same `s_sequence` value, based on the VTAM applid.

Notes

- 1 The default menu used by the system is defined by the `DEFMENU` parameter of the `SYSTEM` statement (see the *Technical Reference*). If the parameter is not specified, a menu called `MENU` is required.
- 2 The `PANEL` definition(s) can be amended so that the format of the Menu display conforms to your Installation standards. For details of the `PANEL` statement and related parameters, see the *Panels, Scripts and Variables* manual.

Batch Administration

Batch Administration enables the creation, modification, or deletion of many configuration statements to take place at once. Using Batch Administration can dramatically reduce the effort required to configure and maintain Session Manager installations.

In order to use Batch Administration, you will need to have an OLA-enabled Session Manager installation. Once you have this you can create custom control files that specify the changes that you want to make, and then use the special Batch Administration jobs and commands to carry out the operations.

For more information on using Batch Administration, refer to the *Online and Batch Administration* manual.

Further information

Information on setting up your configuration can be found in ‘How configuration is achieved’ on page 152. Detailed information on the effects of the different parameters applicable to each configuration statement can be found in the *Technical Reference*.

The *Online and Batch Administration* manual gives a detailed account of how to use the OLA interface.

See also

- ‘Accessing applications’ on page 100.
- ‘National language support’ on page 115

CHAPTER 2

Performing a basic Classic or OLA installation

Installing IBM Session Manager for z/OS is a straightforward process which has been broken down into steps. This chapter is for new customers who are carrying out a Session Manager installation for the first time or for customers who are upgrading to a new version or release.

The subjects covered in this chapter are:

- Product changes – Session Manager 2.2.00 and higher – page 58
- Prerequisites – page 61
- Installing a basic Classic or OLA system – page 62

For other installation/conversion tasks see:

- ‘Installing a Functional Enhancement PTF’ on page 71
- ‘Upgrading to new version or release’ on page 77
- ‘Converting a Classic to an OLA system’ on page 445

Configuring the product to suit your requirements is not treated as part of the installation procedure and is described separately, in ‘Post-installation configuration issues’ on page 85.

Product changes – Session Manager 2.2.00 and higher

This information is only relevant to customers upgrading from a release prior to Session Manager 2.2.00. For these customers it is important that you review all the changes below before proceeding with the product installation. Other customers may skip this topic.

Overview

In releases prior to 2.2.00 Session Manager was shipped in *Classic* mode and customers wishing to use Session Manager in *OLA* mode had to install the product in Classic mode, using SMP/E, then run several jobs to convert this installed product to OLA mode, then upgrade their OLA systems. Any maintenance was also supplied only in Classic mode and again further jobs were required to apply this maintenance to OLA systems. However, all these additional jobs were not under the control of the SMP/E environment.

To simplify the installation and maintenance process for customers running Session Manager in OLA mode, the product is now *supplied* in both Classic and OLA modes. To simplify the installation process for OLA mode, and to allow it to be under the control of the SMP/E environment, several major changes to the product have been required.

Product changes from 2.200 onwards

SMP/E installation process

The SMP/E installation process creates both Classic and OLA datasets. The ISZPALOC, ISZSSEN, ISZVALOC and ISZNALOC jobs are no longer required. The ISZ0EJOB is only required if you need to convert a Classic system to an OLA system (see ‘Converting a Classic to an OLA system’ on page 445).

OLA dataset changes

Most OLA dataset names have been changed so that all low level qualifiers start with ‘.SISZ’ and several have been added. Customer OLA datasets start with ‘.SISZC’ and are supplied empty. Base samples datasets start with ‘.SISZB’ and contain members that can be used for new installations, to easily construct and run a basic system quickly and also as examples for more complex configuration. System datasets start with ‘.SISZS’ and contain members that must be present in the executing system. Some system members (for example, MENU) can be tailored by the customer.

For a list of the OLA datasets see ‘Customer, samples and system configuration datasets’ on page 155. The supplied JCL members have all been updated to incorporate these changes.

The customer UPANEL, customer SPANEL, system ASCRIPT, system SPANEL and system TRANSTAB datasets have been removed.

Classic and OLA configuration changes

Configuration member ISZCON01 is still supplied for Classic systems, and there is now a member called ISZCON02 which is used for OLA systems instead of ISZCON01.

Classic to Classic upgrades

When upgrading Classic systems the ISZCLJOB will copy your existing customer dataset to the new customer dataset. An empty Classic customer dataset is supplied with the product and has a low level qualifier of '.SISZCUST'.

OLA to OLA upgrades

When upgrading OLA systems, the ISZCCJOB will copy your existing pre-2.2.00 customer datasets to the new 2.2.00 customer datasets. If you are upgrading from 2.2.00 or higher, use job ISZC0JOB instead of ISZCCJOB.

Please review the instructions in this manual (see 'Upgrading OLA-enabled to OLA-enabled' on page 81) and also at the top of the ISZCCJOB or ISZC0JOB member; and be aware of the following points:

- The pre-2.2.00 UPANEL and SPANEL datasets have been removed from 2.2.00 and their members will be copied to the CONFIG configuration dataset SISZCCNF by the ISZCCJOB. CONFIG configuration members are invoked by COPY statements, therefore a COPY statement will be required for each member from the pre-2.2.00 UPANEL and SPANEL datasets in each ISZCONxx member. Please update your ISZCONxx members accordingly. See 'Sample menu panels' on page 111.
- The common member COMMON for OLA configuration has been renamed as ISZCOMON. When upgrading your OLA systems, and after the completion of the ISZCCJOB, please review all the ISZCONxx members in your configuration and change the COPY COMMON to COPY ISZCOMON.
- From release 2.2.00 the supplied OLA members all start with 'ISZ' and therefore many have been renamed. For a list of supplied members and their pre-2.2.00 and their 2.2.00 member names see 'Handling upgrades and Functional Enhancement PTFs' on page 89.

If your pre-2.2.00 configuration still contains supplied OLA members then when upgrading your OLA systems, and after the completion of the ISZCCJOB, your OLA datasets will contain both the pre-2.2.00 supplied configuration members and the 2.2.00 supplied configuration members. Please review the OLA datasets and delete any supplied members that are no longer required. Please also review, and if necessary modify, any External Security (for example, RACF) definitions so that they match any supplied members that you are using.

OLA MESSAGE PDSE members

All supplied OLA MESSAGE PDSE members will start with 'ISZM' followed by 4 numeric characters; for example, ISZM0001. However for ease of use the OLA online system will only display and use the last 4 numeric characters; for example, it will display 0001. The BATCH facility will allow both formats as input; for example ISZM0001 and 0001.

OLA CLASS PDSE members

All supplied OLA CLASS PDSE members will start with 'ISZCLS' followed by 2 characters; for example, ISZCLSIM. However for ease of use the OLA online system will only display and use the last 2 characters; for example, it will display IM. The BATCH facility will allow both formats as input; for example, ISZCLSIM and IM. It is strongly recommended that you use OLA to update any CLASS members.

OLA COMMAND PDSE members

All supplied OLA COMMAND PDSE members will start with either 'ISZ' or 'ISZZ' followed by 1 to 5 characters; for example, ISZUPDT and ISZZINIT. However for ease of use the OLA online system will display the COMMAND statement name; for example, it will display BROADCAST for PDSE member ISZZBROA. The BATCH facility will allow both formats as input; for example, BROADCAST and ISZZBROA.

For a full list of COMMAND names and COMMAND PDSE member names see 'Parameters for "File I/O"' in the 'Security considerations' chapter of the *Online and Batch Administration* manual.

Classic and OLA language packs

In a Classic system the default English language pack (LP) is invoked via a COPY ISZLEN statement in each ISZCONxx member. In an OLA system the default English language pack (LP) is invoked via a COPY ISZLEN2 statement in each ISZCONxx member. To load a new LP, add to your ISZCONxx members another COPY ISZLyy statement (for Classic) or COPY ISZLyy2 statement (for OLA), where yy is the 2-character language id for the new language.

User exits

The following subset of user exits is now supplied already assembled and linked, although the source is also provided in case you should wish to modify them:

ISZE00DR linked as ISZEXT00

ISZE09DR linked as ISZEXT09

ISZE21SF linked as ISZEXT21

ISZE22DM linked as ISZEXT22

If you configure your system to use the Multiple Exit Driver (ISZE00DR) all available exits will be loaded at initialization of your system. The above list of supplied exits that are already assembled and linked should be reviewed and any exits not required should be removed or renamed.

Prerequisites

This topic is covered in the IBM *Program Directory*.

Installing a basic Classic or OLA system

Installing Session Manager involves several steps, and this chapter of the manual is broken down into individual sections, one for each activity required. Session Manager may be installed, generated and activated during normal system running.

The subjects covered in this section are:

- ‘Step 1: Preliminary checks’ on page 62
- ‘Step 2: Installation’ on page 62
- ‘Step 3: APF authorization’ on page 62
- ‘Step 4: VTAM alterations’ on page 63
- ‘Step 5: Configuration file customization’ on page 65
- ‘Step 6: Starting the system – initiating Session Manager’ on page 66
- ‘Step 7: Signing on to Session Manager’ on page 67

Step 1: Preliminary checks

Before installing Session Manager carry out these environment checks:

- 1 Review the prerequisites set out in the *Program Directory*.
- 2 Ensure that Session Manager will have a dedicated address space in which to run.

This should not present a problem, but you may need to review your loading and scheduling prior to installing and activating Session Manager. In particular, the dispatching priority of the address space in which Session Manager is running should be greater than any applications accessed through it.

Step 2: Installation

Follow the installation procedure given in the *Program Directory*. The product is shipped in Classic and in Online and Batch Administration (OLA) format. If after installing the Classic system, you later decide to execute in OLA mode, you can convert your Classic system to an OLA format (see ‘Converting a Classic to an OLA system’ on page 445).

Step 3: APF authorization

The load library must be APF-authorized to enable Session Manager to run and become non swap-able.

To APF authorize the load library, *one* of the following methods could be used:

- If all the datasets in the LNKST concatenation are treated as APF authorized, (LNKAUTH=LNKST), add *hlq*.SISZAUTH, where *hlq* is the appropriate high-level qualifier, to LNKSTxx. Perform an IPL to make this effective.

or

- Copy the contents of *hlq*.SISZAUTH to an existing APF authorized LNKST library.

or

- Update the member IEAAPFxx with hlq.SISZAUTH. Add a steplib for hlq.SISZAUTH to the startup deck ISZCSTRT. Perform an IPL to make this effective.

or

- Copy the contents of hlq.SISZAUTH to an existing library in IEAAPFxx. Add a steplib for that load library to the startup deck ISZCSTRT.

or

- Use the APF authorization command SETPROG.

Note The SETPROG command is a temporary solution until the z/OS system is re-IPL'ed.

Step 4: VTAM alterations

There are two changes that are required in the VTAM environment. These are:

- 1 To define Session Manager as an application to VTAM, together with minor node entries for all applications not capable of running parallel sessions, such as CICS.
- 2 To define logmode table entries that are necessary for Session Manager to function correctly.

It is recommended that you read 'Accessing applications' on page 100 before completing this step.

Modifying VTAM definitions

It is necessary to define Session Manager as an application to VTAM. To do this, set up an application major node, or use the sample member ISZVAPPL either as it stands, or as a guide to defining an installation-specific major node. The VTAM definition macro given below is an example of how the required application major node is defined:

```
ISZ  VBUILD TYPE=APPL
```

The following two minor node entries should also be defined:

```
ISZSMGR  APPL  ACBNAME=ISZSMGR,AUTH=(ACQ,PASS,NVPAGE),EAS=nnn,
          PARSESS=YES,VPACING=0
```

```
ISZ001  APPL  ACBNAME=ISZ001,AUTH=(ACQ,NVPAGE),EAS=nnn,
          PARSESS=YES,VPACING=0,MODETAB=ISZVMODE
```

In this example, the name ISZSMGR is the name of the Session Manager system ACB; that is, the ACB that will be used for logging on to Session Manager itself, and will be specified on the ACB parameter of the Session Manager SYSTEM statement. ISZ001 defines the default session ACB that Session Manager will use to establish sessions with applications, and will be defined on the SESCAB parameter of the Session Manager SYSTEM statement. See 'Step 5: Configuration file customization' on page 65 for details about updating the Session Manager configuration.

The EAS parameter for Session Manager should be set to the maximum number of terminals ever likely to be signed on to Session Manager simultaneously (default 404), and the ACBNAME should be the same as that specified on the ACB parameter of the SYSTEM statement. For ISZ001, the EAS parameter should be set to the maximum number of parallel sessions that Session Manager is ever likely to be running using this ACB, and the ACBNAME should be the same as that specified on the SESACB parameter of the SYSTEM statement.

Minor node entries should also be made for all other applications not capable of running parallel sessions, such as CICS. See 'Accessing applications' on page 100, to determine your exact requirements. As a minimum, Session Manager requires a parallel session ACB (the ACB defined on the SESACB parameter of the SYSTEM statement) in addition to its logon ACB. Define these to start, and then define others as required. For example:

```
ISZ002 APPL ACBNAME=ISZ002,
      AUTH=(ACQ,NVPACE),EAS=nn,PARSESS=YES,MODETAB=ISZVMODE
      SRBEXIT=YES or AUTHEXIT=YES
      ... and so on ...
ISZ050 APPL ACBNAME=ISZ050,
      AUTH=(ACQ,NVPACE),EAS=nn,PARSESS=YES,MODETAB=ISZVMODE
      SRBEXIT=YES or AUTHEXIT=YES
```

The EAS parameter is the estimated number of concurrent sessions which will be activated on the ACB. This should be adjusted as required. If you underestimate this value, it will not prevent extra sessions being started.

The ISZVMODE parameter should be coded so that if any of the ACBs are to be used with a terminal that has extended attributes, appropriate sessions will be established. The logmode table ISZVMODE is supplied on the distribution media with the entries required for such terminal types. This is discussed further in 'ISZVMODE MODETAB' below.

Session Manager supports VTAM FASTPATH. This reduces the amount of validation performed by VTAM and increases the throughput. Installations wishing to use FASTPATH should code SRBEXIT YES on the APPL statement for each minor node in the application major node for Session Manager. This includes the Session Manager ACB, ISZSMGR, and all the dummy ACB definitions.

The PARSESS parameter is coded on the logical terminal to allow multiple sessions to exist between that logical terminal (ACB) and the application.

Note Change these names (ISZnnn) to conform to any Installation-specific naming standards to avoid duplication of ACB names.

See the IBM *z/OS Communications Server: SNA Resource Definition Reference* manual for a full description of any of the VTAM parameters.

Activating VTAM resources

The VTAM resources must be made available to VTAM once all the necessary Installation specific changes have been made.

The modified definitions must be copied to a partitioned data set that is accessible to VTAM and then varied active in the normal way.

**ISZVMODE
MODETAB**

A Logmode Entry Table is supplied on the distribution media in both source and executable form. The assembled and linked table is called ISZVMODE (supplied in library .SISZAUTH). The source file is called ISZMODE (supplied in library .SISZCONF).

ISZVMODE contains entries for terminals such as the 3279, 3180 and 3193 models which support extended attributes. Full details of the table can be found in the section on setting up applications in the *Technical Reference*.

Ensure that the ISZVMODE module is stored in a library available to VTAM.

The supplied table can be amended to suit the naming conventions used at your Installation. Alternatively, if your Installation has its own logmode table, then these entries may just be added to your existing table and reassembled.

**Assembling the
supplied
logmode table
(optional)**

Session Manager is supplied with the source of the logmode table which may be assembled if it is preferred to do it this way.

Step 5: Configuration file customization

When Session Manager has been successfully installed, and any required changes have been made to VTAM and the operating environment, the Session Manager configuration should be set up to your Installation's requirements.

For a Classic system:

As supplied, the control statements (which specify the terminals, users and applications in the system) are read from a Classic configuration – that is, all configuration definitions are stored in members of PDS(s) allocated to the DDNAME of CONFIG.

For an OLA system:

As supplied, the control statements (which specify the terminals, users and applications in the system) are read from an OLA configuration. The configuration data is stored in several datasets and a specific DDNAME is associated with each statement type (APPL, PROFILE, USER, and so on).

Configuration members ISZCON01 (for Classic, in library .SISZCONF) and ISZCON02 (for OLA, in library .SISZSCNF) supplied on the Session Manager distribution media contain many examples of the various control statements and parameters required. See 'Performing a basic configuration' on page 151, for further details of possible changes that can be made to tailor this configuration file for your Installation.

Before starting the system for the first time, the ACB name to be used for logging on to the Session Manager system must be specified on the SYSTEM statement. You should also specify the default session ACB to be used for sessions.

First copy the member(s) containing these statements to the customer dataset(s). For Classic, copy ISZCON01 from library .SISZCONF to library .SISZCUST. For OLA, copy ISZCON02 from library .SISZSCNF to library .SISZCCNF and copy ISZSYS02 from library .SISZSSYS to library .SISZCSYS.

Rename the ISZCONnn member just copied as ISZCONtt (where tt is a unique suffix to be used for testing). For OLA, also rename the copied ISZSYS02 as ISZSYSStt.

You should also change the ACB and SESACB parameters to refer to ACB names that you will be using for testing. For Classic, the SYSTEM statement will be in ISZCONtt. For OLA, The SYSTEM statement will be in ISZSYSStt.

Note In an OLA environment the SYSTEM statement should normally only be updated by using the OLA system. For this reason the OLA SYSTEM members explicitly state 'DO NOT EDIT'. However at this point in the installation there is no active system, so the SYSTEM statement should be carefully updated manually using an editor such as TSO/ISPF.

Only the ACB and SESACB should be changed, and the format of the file must be maintained as supplied.

Step 6: Starting the system – initiating Session Manager

Alter the sample startup (in ISZCSTRT for Classic, in ISZCOLA for OLA – both in library .SISZCONF) to suit your Installation, change the startup parameters to reference CONFIG tt (where tt is the suffix you are using for testing), and place in an appropriate PROCLIB, either SYS1.PROCLIB or one of its concatenations. It can then be initiated as a started task using the MVS START (S) command.

Return codes

The following return codes may be issued:

Return code	Reason
0	Session Manager completed OK
8	INITIAL_CMD script was not found
16	INITIAL_CMD script suffered a logic error
256	Session Manager load module is not APF-authorized
260	Session Manager configuration error
264	Session Manager spool manager failed to initialize
268	An instance with the same LOCALNODE value is already active in the SYSPLEXGROUP. See message ISZ0185E or ISZ0186E for more information.
272	Session Manager Controller completed OK after successful SWITCHPLX.
276	Session Manager Instance completed OK after successful SWITCHPLX.
284	Instance specified on STANDBY already has an active standby.
288	STANDBY parameter specifies local node; that is, the standby is trying to stand by for itself.
292	A SYSPLEXTYPE N instance has a STANDBY parameter specified on the SYSTEM statement. This configuration error is detected at Session Manager start up, causing immediate automatic close-down. Message ISZ0450E will be issued.

Return code Reason (continued)

n INITIAL_CMD script returned this value. Note that this script can return any return code, including any of the above.

Note It is suggested that INITIAL_CMD script authors should not return these values.

Terminating Session Manager

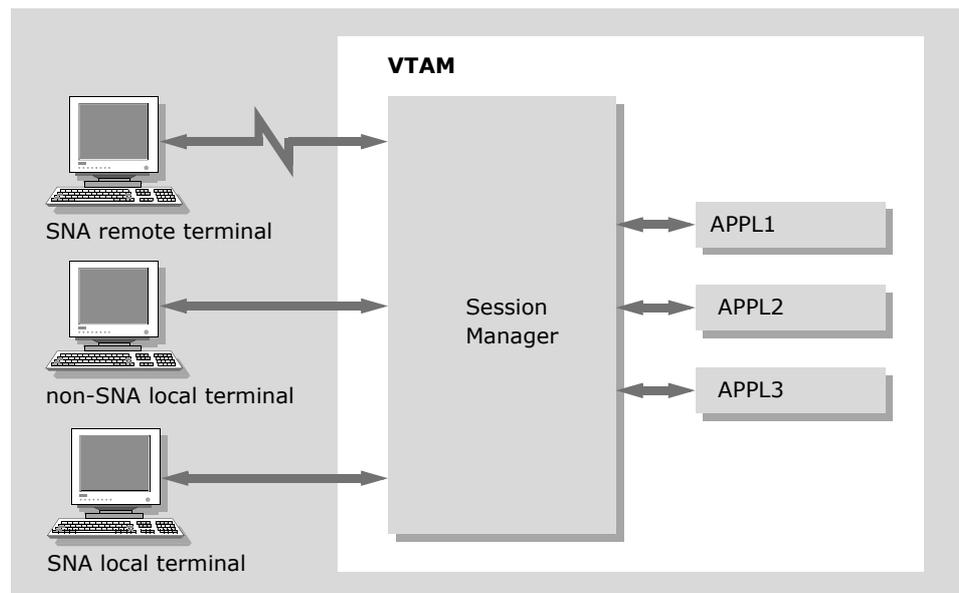
To terminate Session Manager, the CLOSEDOWN command should be issued.

Step 7: Signing on to Session Manager

Since Session Manager runs as a normal VTAM application, users establish contact with it simply through a normal VTAM logon. The user at a terminal may type the logon from the VTAM MSG10 screen.

If the Installation requires one or more terminals to be automatically logged on to Session Manager, the LOGAPPL operand may be coded on the appropriate LU and LOCAL VTAM macros for the relevant terminals. Automatic logon can also be achieved using the VTAM 'VARY NET,LOGON' operator command.

When the logon has been successfully processed, the Session Manager signon panel is displayed, although the main menu panel may be displayed if so configured.



The following information is included to help you to logon to Session Manager directly after installation. For an overview of security and using signon panels, see 'Implementing security' on page 99.

Logon data from the terminal

Once Session Manager has been installed, various types of security – including using an ESM to validate users – can be implemented, along with customized signon panels. For details, see 'Implementing security' on page 99.

Users who would normally enter the system using the signon panel can in fact bypass this display by entering their userid (and password if required) as VTAM logon data when logging on to Session Manager. For example:

```
LOGON APPLID=ISZSMGR,DATA=userid/password
```

or

```
LOGON APPLID(ISZSMGR) DATA(userid/password)
```

depending on the Installation's logon command format.

A 'new password' and 'new profile' may also be included in the logon data. The new password is useful only for Installations using a security exit. For example:

```
LOGON APPLID=ISZSMGR,DATA=userid/password/newpassword/newprofile
```

or, if just the new profile were required:

```
LOGON APPLID=ISZSMGR,DATA=userid///newprofile
```

The users defined in the supplied sample configuration file are:

Member name	Description
<i>Classic:</i>	
ISZCUSER	Sample Users ISZUSER and ISZSYSAD (in library .SISZCONF)
<i>OLA:</i>	
ISZSYSAD	Sample Administrator (in library .SISZBUSR)
ISZUSER	Sample User (in library .SISZBUSR)

Specifying a different terminal name

If a different terminal definition to that normally associated with a terminal is required, a terminal definition name can be passed in the logon data. For example, if SIGNON NO is in effect:

```
LOGON APPLID=ISZSMGR,DATA=///termname
```

If the terminal name does not match a specific name defined on a TERMINAL statement, a match may be found with one defined generically. If no match is found, the default terminal definitions as specified by the SYSTEM statement are used.

Following the terminal name, four fields are available in which user defined data may be supplied. The fields are unique to each user and each field has a maximum length of 20 bytes. The contents may be modified by various processes. The Signon validation exit point of the Session Manager user exit (E21) may change the values. In addition, these variables may be used/modified on panel definitions, in scripts, or for hardcopy variable substitution. Should a user disconnect, the values are preserved until the reconnect occurs.

For example:

```
LOGON APPLID=ISZSMGR,DATA=userid/password///ucva-value
/ucvb-value/ucvc-value/ucvd-value
```

You should now have a basic Session Manager system up and running. Confirm that you can log on to it using your test ACB and sign on to it using the sample userids described in 'Logon data from the terminal' on page 67.

You are now ready to proceed with configuring the system by moving on to either 'Post-installation configuration issues' on page 85 or 'Upgrading to new version or release' on page 77.

CHAPTER 3

Installing a Functional Enhancement PTF

This section is for customers who are installing a Functional Enhancement PTF.

The subjects covered are:

- 'Introduction' on page 72
- 'Applying a PTF in a Classic system' on page 73
- 'Applying a PTF to an OLA system' on page 75

For other installation/conversion tasks see:

- 'Performing a basic Classic or OLA installation' on page 57
- 'Upgrading to new version or release' on page 77
- 'Converting a Classic to an OLA system' on page 445

Configuring the product to suit your requirements is not treated as part of the installation procedure and is described in 'Post-installation configuration issues' on page 85.

Introduction

Functional Enhancement PTFs are supplied in Classic format and in OLA format. Classic and OLA datasets are updated via the SMP/E process.

Applying a PTF in a Classic system

To apply a Functional Enhancement PTF to a Classic system proceed as follows:

Installing the PTF

- 1 Back up your existing datasets.
- 2 Apply the PTF to the Classic environment using the SMP/E process.
- 3 If required, add Language Packs (LPs).
- 4 A sample procedure for the new Session Manager started task can be found in member ISZCSTRT (in library .SISZCONF) if changed in the supplied PTF. Tailor this sample job to your Installation's requirements.
- 5 In your test environment, start the sample ISZCSTRT started task. Check that the new Session Manager system starts up successfully and that *the configuration is error free*.
- 6 Test the new Session Manager system to ensure that the product has been installed correctly.
- 7 Shut down the new Session Manager system when you have verified that everything is working as expected.

Customizing Session Manager

- 1 If it has changed, copy the supplied ISZCON01 configuration member to your customer dataset and then, if appropriate, rename it to ISZCONxx (where xx is the unique identifier for your current Session Manager configuration). Changed members should be saved in the customer configuration library .SISZCUST.
- 2 Tailor the ISZCONxx configuration member to your Installation's requirements (changed members should be saved in the customer configuration library .SISZCUST):
 - a Review any changes (COPY statements, SYSTEM statement parameters *excluding* ACB, and so on) you made to the existing ISZCONxx member and, if they are still required, re-apply them to the new ISZCONxx member.
 - b Add Language Packs (LPs):
 - The default LP is defined in the COPY ISZLEN statement, where EN is the language id for English.
 - To load a new LP, add an additional COPY ISZLyy statement, where yy is the 2-character language id for the new language, to your ISZCONxx member.
 - c Check supplied member ISZ£READ for any other changes that you may need to make to your customer configuration definitions.
- 3 Review any changes you made to your existing Panels, Scripts and Messages and, if the changes are still required, re-apply them to the new set of Panels, Scripts and Messages. Review 'Handling upgrades and Functional Enhancement PTFs' on page 89 for information on Functional Enhancement PTFs in relation to supplied samples (configuration, exit and JCL samples). See also 'Storage requirements' on page 86.

- 4 To allow you to test your applications, review the FROM parameter on any ACB RANGE statements and change the values to correspond to the VTAM acbnames in your test environment.
- 5 In your test environment, start the new customized Session Manager system and check that *the configuration is error free*.
- 6 Test the new customized Session Manager system to ensure that the product has been correctly configured.
- 7 Shut down the new customized Session Manager system when you have verified that everything is working as expected.

Putting Session Manager into production

After you have successfully customized the new Session Manager system (see page 73), complete the following steps to put the new release into production.

- 1 In the ISZCONxx configuration member, change the ACB ISZSMGR and SESACB ISZ001 parameters on the SYSTEM statement to match the VTAM acbnames in your production environment.
- 2 To allow you to use your applications, review the FROM parameter on any ACB RANGE statements and change the values to correspond to the VTAM acbnames in your production environment.
- 3 At a convenient time, shut down your existing production Session Manager system and change the procedure that starts this system to reference the new release libraries.
- 4 Re-start the production Session Manager system.

Applying a PTF to an OLA system

To apply a Functional Enhancement PTF to an OLA system proceed as follows:

Installing the PTF

- 1 Back up your existing datasets.
- 2 Apply the PTF to the OLA environment using the SMP/E process.
- 3 If required, add Language Packs (LPs).
- 4 A sample procedure for the new OLA-enabled Session Manager started task can be found in member ISZCOLA (in library .SISZCONF). Tailor the JCL in this sample procedure to your Installation's requirements.
- 5 In your test environment, start the tailored ISZCOLA started task. Check that the OLA-enabled Session Manager system starts up successfully and that *the configuration is error free*.
- 6 Test the new OLA-enabled Session Manager system to ensure that the product has been correctly configured.
- 7 Shut down the new OLA-enabled Session Manager system when you have verified that everything is working as expected.

Customizing Session Manager

From Session Manager 2.2.00 the OLA configuration is supplied in member ISZCON02.

- 1 If it has changed, copy the supplied ISZCON02 (in library .SISZSCNF) configuration member to your customer dataset and then, if appropriate, rename it to ISZCONxx (where xx is the unique identifier for your current Session Manager configuration). Changed members should be saved in the customer configuration library .SISZCCNF.
- 2 Tailor the ISZCONxx configuration member to your Installation's requirements (changed members should be saved in the customer configuration library .SISZCCNF):
 - a Review any changes (COPY statements, SYSTEM statement parameters *excluding* ACB, and so on) you made to the existing ISZCONxx member and, if they are still required, re-apply them to the new ISZCONxx member.
 - b Add Language Packs (LPs):
 - The default LP is defined in the COPY ISZLEN2 statement, where EN is the language id for English.
 - To load a new LP, add an additional COPY ISZLyy2 statement, where yy is the 2-character language id for the new language, to your ISZCONxx member.
 - c Check supplied member ISZEREAD for any other changes that you may need to make to your customer configuration definitions.

- 3 Review 'Handling upgrades and Functional Enhancement PTFs' on page 89 for information on Functional Enhancement PTFs in relation to supplied samples (configuration, exit and JCL samples). See also 'Storage requirements' on page 86.
- 4 To allow you to test your applications, review the FROM parameter on any ACB RANGE statements and change the values to correspond to the VTAM acbnames in your test environment.
- 5 In your test environment, start the new customized Session Manager system and check that *the configuration is error free*.
- 6 Test the new customized Session Manager system to ensure that the product has been correctly configured.
- 7 Shut down the new customized Session Manager system when you have verified that everything is working as expected.

Putting Session Manager into production

After you have successfully customized the new Session Manager system (see page 75), complete the following steps to put the new release into production:

- 1 Using OLA, change the ACB ISZSMGR and SESACB ISZ001 parameters on the SYSTEM statement to match the VTAM acbnames in your production environment. (Reply N when the Immediate activation of change? prompt is displayed.)
- 2 To allow you to use your applications, review the FROM parameter on any ACB RANGE statements and, using OLA, change the values to correspond to the VTAM acbnames in your production environment. (Reply N when the Immediate activation of change? prompt is displayed.)
- 3 At a convenient time, shut down your existing production Session Manager system and replace the procedure that starts this system with the procedure that starts the new OLA-enabled Session Manager system.
- 4 Re-start the production Session Manager system.

CHAPTER 4

Upgrading to new version or release

This chapter explains how an existing customer can upgrade to a new version or release of IBM Session Manager for z/OS. You must first have completed a basic installation, as described in ‘Performing a basic Classic or OLA installation’ on page 57.

CAUTION Before proceeding, make your existing configuration secure by backup.

The subjects covered in this chapter are:

- ‘Upgrading Classic to Classic’ on page 78
- ‘Upgrading OLA-enabled to OLA-enabled’ on page 81

For other installation/conversion tasks see:

- ‘Performing a basic Classic or OLA installation’ on page 57
- ‘Installing a Functional Enhancement PTF’ on page 71
- ‘Converting a Classic to an OLA system’ on page 445

Configuring the product to suit your requirements is not treated as part of the installation procedure and is described in ‘Post-installation configuration issues’ on page 85.

Upgrading Classic to Classic

In a Classic configuration, all configuration definitions are stored in members of PDS(s) allocated to the DDNAME of CONFIG. To upgrade such a configuration, any customization applied to the current version must be reapplied to the new version. The new version must then be tested, possibly in parallel with the Production system. Finally, when testing is complete, the new version must be put into production. Jobs are supplied to facilitate this process. The following sections describe and explain a suggested technique to perform the upgrade.

Installing Session Manager

The new version should first be installed as described in ‘Performing a basic Classic or OLA installation’ on page 57.

You should then have a startup procedure (ISZCSTRT) which will launch a basic Session Manager system using ISZCONtt (where tt is a unique suffix for testing) based on the supplied member ISZCON01.

The customer dataset (library .SISZCUST) should at this stage contain ISZCONtt only.

Customizing Session Manager

The next step is to incorporate customization applied to the current version into the new version.

- 1 If you have not yet split your customer configuration definitions from system configuration definitions, then do so now:

Copy customer configuration definitions (that is, installation-specific definitions such as USER, PROFILE, SYSTEM, APPL, plus any new or modified panels, scripts, and so on) into the customer configuration dataset supplied with the new version (library .SISZCUST).

If you already maintain customer configuration data separately from system configuration definitions, then tailor and run the supplied job ISZCLJOB (in library .SISZCONF) to copy your definitions into the new customer configuration dataset.

- 2 Tailor ISZCONtt to your installation’s requirements:
 - a Review any changes (COPY statements, SYSTEM statement parameters excluding ACB) made to the existing ISZCONxx member used in Production and, if still required, reapply them to ISZCONtt.
 - b If required, add Language Packs (LPs):
 - the default LP is defined in the COPY ISZLEN statement where EN is the language id for English.
 - to load a new LP, add an additional COPY ISZLyy statement (where yy is the 2-character language id for the new language) to your ISZCONtt member. The newly supplied LP should also be added to the CONFIG DD concatenation.

- 3 For testing purposes, particularly if you plan to perform testing in parallel with running the Production system, you may wish to review the ACB names used by Session Manager for application sessions (for example, SESACB on the SYSTEM statement, RANGE statements, and so on) and change the values to correspond to VTAM ACB names in your test environment. Remember that you will need to change them back again to go into Production.
 - 4 Review ‘Handling upgrades and Functional Enhancement PTFs’ on page 89 for information on upgrades in relation to supplied samples (configuration, exit and JCL samples). See also ‘Storage requirements’ on page 86.
 - 5 Review any startup parameters in the current Session Manager startup procedure and, if required, apply them to your ISZCSTRT procedure.
Do not at this stage change the CONFIG parameter – it should still refer to ISZCONtt for testing.
 - 6 If you are upgrading from Session Manager release 1.3.05 or higher to a later release and you are using the Virtual Terminal Masking (VTM) facility, you must run the supplied job ISZVCOPY (in library .SISZCONF) to copy the current VTM PDSE to the new VTM PDSE.
If you don’t use the VTM facility, comment out the DD definition for VTM in your ISZCSTRT procedure.
 - 7 The NODE PDSE is needed in order to run the Sysplex user interface. There is no need to copy this from the previous release. If you don’t use the Sysplex user interface, comment out the DD definition for NODE in your ISZCOLA procedure.
 - 8 Check the supplied member ISZ£READ (in library .SISZCONF) for any other changes you may need to make to your customer configuration definitions for the new version.
 - 9 In your test environment, start the customized system using your ISZCSTRT procedure. Check that the configuration is error free and that any required exits are loaded.
- Note** Some commonly used exits are now supplied pre-assembled and linked, so will be available automatically (see ‘User exits’ on page 60 for details). If you use the Session Manager exit driver ISZEXT00, it will automatically load these pre-linked exits, so you may wish to delete from the load library or rename any that you don’t use. If you have made user modifications to any exits, you may wish to incorporate your changes into the new versions and reassemble and link them into the new load library.
- 10 Test the system to ensure that the product has been correctly configured.
 - 11 Shut down the system when you have verified that everything is working as expected.

Putting Session Manager into production

When you are ready to put the new version into production, you will first need to reverse any ACB name changes you made in Step 3 above so that only production ACB names are referenced, then copy or rename ISZCONtt as ISZCONxx (where xx is the suffix to be used for the Production system). Change the startup parameters in your ISZCSTRT procedure to specify CONFIG xx and start up the system again.

Upgrading OLA-enabled to OLA-enabled

In an OLA configuration, configuration data is stored in several datasets and a specific DDNAME is associated with each statement type (APPL, PROFILE, USER and so on). To upgrade such a configuration, any customization applied to the current version must be reapplied to the new version. The new version must then be tested, possibly in parallel with the Production system. Finally, when testing is complete, the new version must be put into production. Jobs are supplied to facilitate this process. The following sections describe and explain a suggested technique to perform the upgrade.

Installing Session Manager

The new version should first be installed as described in ‘Performing a basic Classic or OLA installation’ on page 57.

You should then have a startup procedure (ISZCOLA which will launch a basic Session Manager system using ISZCONtt (where tt is a unique suffix for testing) based on the supplied member ISZCON02.

Customizing Session Manager

The next step is to incorporate customization applied to the current version into the new version:

- 1 Copy your existing OLA-enabled customer configuration datasets to the newly allocated set of OLA-enabled configuration datasets. The product is supplied with a set of empty customer datasets with a low level qualifier that starts with .SISZC. A sample job to do this can be found in member ISZCCJOB (for upgrades from pre-2.2.00 systems) or job ISZCOJOB (for upgrades from 2.2.00 and higher systems) (in library .SISZCONF). Review the instructions in ISZCCJOB or ISZCOJOB carefully and amend the JCL accordingly.
- 2 Member COMMON has been renamed as ISZCOMON from Session Manager release 2.2.00, so you will need to edit any ISZCONxx members copied in Step 1 to change COPY COMMON to COPY ISZCOMON.

From Session Manager 2.2.00, panel definitions no longer reside in OLA-enabled datasets, so ISZCCJOB will have copied any found in the UPANEL and SPANEL datasets to the SISZCCNF configuration dataset. You will need to edit your ISZCONxx members, including ISZCONtt, to add a COPY statement, after the COPY ISZCOMON, for each panel copied in order for Session Manager to pull in these panel definitions at startup.

Warning: If you have your own versions of supplied panels, such as SIGNON or MENU, and haven’t changed the PANEL names, you will get ‘statement already exists’ messages when you come to start up your system and the new supplied version will be used in preference. If you wish to continue using your versions, you should change the PANEL names on your modified versions (now in .SISZCCNF) to prevent the duplicates, and later, using OLA, change any references to those panels in SYSTEM, PROFILE, USER statements so that they refer to the new names.

- 3 Review any changes made to your existing ISZCONxx member (for example, OPTION statement changes other than EXIT, configuration logic to set gc_acb_prefix and gc_acb_suffix), and apply to your ISZCONtt member.

Note Before you can start testing in earnest, you will first need to bring up the system to perform some OLA updates, which we suggest you do from the supplied user ISZSYSAD. Since the signon to ISZSYSAD is likely to be rejected by your ESM, you should bring up Session Manager with no exits for this initial run.

- 4 Review 'Handling upgrades and Functional Enhancement PTFs' on page 89 for information on upgrades in relation to supplied samples (configuration, exit and JCL samples). See also 'Storage requirements' on page 86.
- 5 If you are upgrading from Session Manager release 1.3.05 or higher to a later release and you are using the Virtual Terminal Masking (VTM) facility, you must run the supplied job ISZVCOPY (in library .SISZCONF) to copy the current VTM PDSE to the new VTM PDSE.

If you don't use the VTM facility, comment out the DD definition for VTM in your ISZCOLA procedure.

- 6 The NODE PDSE is needed in order to run the Sysplex user interface. There is no need to copy this from the previous release. If you don't use the Sysplex user interface, comment out the DD definition for NODE in your ISZCOLA procedure.
- 7 Check the supplied member ISZ£READ (in library .SISZCONF) for any other changes you may need to make to your customer configuration definitions for the new version.
- 8 Review any startup parameters in the current Session Manager startup procedure and, if required, apply them to your ISZCOLA procedure. At this stage do not change the CONFIG parameter - it should still refer to ISZCONtt for testing, and do not code the EXIT parameter (see note in Step 3 above).
- 9 For testing purposes, particularly if you plan to perform testing in parallel with running the Production system, you may wish to review the ACB names used by Session Manager for application sessions (for example, SESACB on the SYSTEM statement, RANGE statements, and so on) and change the values to correspond to VTAM ACB names in your test environment.

To do this, bring up your test Session Manager system using your ISZCOLA procedure. Sign on as ISZSYSAD (see 'Step 6: Starting the system – initiating Session Manager' on page 66) and use OLA to change the ACB and SESACB parameters on the ISZSYSxx definition (where xx is the suffix used for the Production system) to ACB names in your test environment, plus any other statements affected. Remember that you will need to change them back again to go into Production.

- 10 If you are upgrading a pre 1.3.05 Session Manager release and are running multiple instances then you may also want to split the SYSTEM members ISZSYSnn into ISZSYSn and ISZSYSnn. The ISZSYSn member should contain all the common SYSTEM parameters for all the instances and the individual ISZSYSnn members should contain the SYSTEM parameters unique to that instance.

To do this, copy one of your ISZSYSnn members as IZSYSM. Update the individual ISZSYSnn members to delete all parameters explicitly specified with the exception of those unique to that instance – typically ACB, LOCALNODE, SESACB and TCP. Update ISZSYSM to delete just those same parameters.

Note ISZSYSM is a reserved name used by Session Manager configuration processing. Settings specified on SYSTEMM will override system default values, and they will in turn be overridden by settings specified on the individual SYSTEMnn member.

Shut down the Session Manager test system when you have made all the required OLA updates.

- 11** Change the startup parameters in your ISZCOLA procedure to refer to ISZCONxx (where xx is the suffix for the Production version). If you run Session Manager user exits, code the EXIT parameter appropriately either in the startup JCL or in the OPTION statement in ISZCONxx. If you have made user modifications to any exits, you may first wish to incorporate your changes into the new versions and reassemble and link them into the new load library.

Note Some commonly used exits are now supplied pre-assembled and linked, so will be available automatically (see ‘User exits’ on page 60 for details). If you use the Session Manager exit driver ISZEXT00, it will automatically load these pre-linked exits, so you may wish to delete from the load library or rename any that you don’t use. If you have made user modifications to any exits, you may wish to incorporate your changes into the new versions and reassemble and link them into the new load library.

- 12** Restart the system to run using ISZCONxx, check that the configuration is error free and that any required exits are loaded, and test to ensure that the product has been correctly configured.
- 13** Shut down the system when you have verified that everything is working as expected.

Putting Session Manager into production

When you are ready to put the new version into production, you will first need to use OLA to reverse any ACB name changes you made in Step 9 above so that only production ACB names are referenced, then shut down and restart.

CHAPTER 5

Post-installation configuration issues

This chapter introduces some important IBM Session Manager for z/OS concepts, and provides advice on the topics you should be considering when configuring your Installation.

Once you have read this chapter, move on to ‘Performing a basic configuration’ on page 151.

The subjects covered in this chapter are:

- ‘Storage requirements’ on page 86
- ‘Handling upgrades and Functional Enhancement PTFs’ on page 89
- ‘Implementing security’ on page 99
- ‘Accessing applications’ on page 100
- ‘Defining System, Profile and User settings’ on page 103
- ‘Designing panels and menus’ on page 111
- ‘National language support’ on page 115
- ‘Using scripts with applications’ on page 118
- ‘User Exits and the Multiple Exit Driver’ on page 124
- ‘Setting TCP/IP support (if required)’ on page 125
- ‘Networking and Sysplex considerations (if required)’ on page 127
- ‘Setting Audit GDG dataset support (if required)’ on page 133
- ‘Implementing Automatic Restart Manager (if required)’ on page 135
- ‘Using multiple Session Manager instances – networking’ on page 137
- ‘Setting up OLA for single and multiple Session Manager instances’ on page 141
- ‘Debugging and problem diagnosis facilities’ on page 147
- ‘Access from CICS to Session Manager’ on page 149

Storage requirements

The Session Manager program, ISZSMGR, is approximately 3000K in size.

On top of this fixed amount, Session Manager requires varying amounts of dynamic storage (GETMAIN) depending on the configuration in use. In order to calculate the approximate size required, the following section is included to help you calculate the amount of dynamic storage likely to be required.

Estimating the dynamic storage requirement

- 1 At least 400K of permanently allocated dynamic storage which is used for the system; the size depends on the size of the configuration file.
- 2 The sample configuration (ISZCON01 for Classic, ISZCON02 for OLA) as supplied on the installation media is approximately 6500K. This consists of:
 - 4500K standard definitions
 - 2000K default Language Pack

Note Allow 2000K for each additional Language Pack.

- 3 14K of dynamic storage for each terminal/user signed on to Session Manager, including those which are currently disconnected.

If using the Shared User Facility (SHARE, SHAREDISC or SHARESESS), count users sharing a userid as separate users.

Note If the INPUTEXIT parameter has been specified on the SYSTEM configuration statement, and the module has been successfully loaded, then an additional 1296 bytes *plus* the number of bytes specified on the EXITWALEN parameter will be required for each terminal/user.

- 4 3K of dynamic storage for each active session, including sessions which are associated with a Menu screen that is currently disconnected. Plus an additional 4K per session if running MISER.

If using the Shared User Facility to share sessions (SHARESESS), count a separate session for each user sharing a session.

Note If the OUTPUTEXIT parameter has been specified on the SYSTEM configuration statement, and the module has been successfully loaded, then an additional 1296 bytes *plus* the number of bytes specified on the EXITWALEN parameter will be required for each active application session.

- 5 5K for each internal trace task initiated, but not for data tracing.
- 6 4K of dynamic storage for each open ACB.
- 7 10K of dynamic storage for each terminal or user that receives a Session Manager broadcast. Storage reduces to 3K when the Broadcast panel appears and this final 3K is freed when the user exits from the Broadcast panel.
- 8 If MAXSTOR is specified on the configuration OPTION statement to improve performance, the total dynamic storage is significantly increased. It is possible that MAXSTOR may double the dynamic storage required.

The storage amounts given in numbers 1, 3, 4 and 6 above are 'average' figures which will vary according to the configuration options chosen.

After using these figures to calculate an approximate total storage requirement, it is advisable to round the figure up rather than down: better to have too much storage available than to find that Session Manager cannot initiate sessions because of storage shortages.

When any panel is displayed it uses between 1 and 2K of storage. This storage is released on quitting from the panel. However, with the Query panel, escape is permitted and in this instance the storage used by the panel is not released.

Examples of storage requirements

Example 1

An Installation requires 100 users to have an average of 4 active sessions each:

Function	Size
Session Manager itself	3000K
Permanent allocation	400K
Terminals logged on (100)	1400K
Active session (400) MISER Yes	2800K
Default Language Pack	2000K
Remaining configuration	4500K
Broadcast to all users (100)	1000K
Total size	15100K

- 1 If MAXSTOR is set to N, then define STORLIM as 16M.
- 2 If MAXSTOR is set to Y, then set STORLIM to 32M.

Example 2

An Installation has 50 users with an average of 2 active sessions each. An additional Language Pack has been loaded, and allowance is made for 10 internal trace tasks to be simultaneously active:

Function	Size
Session Manager itself	3000K
Permanent allocation	400K
Terminals logged on (50)	700K
Active session (100) MISER No	300K
Default Language Pack	2000K
Additional Language Pack	2000K
Remaining configuration	4500K
Broadcast to all users	500K
Trace task storage (10)	50K
Total size	13450K

- 1 If MAXSTOR is set to N, then define STORLIM as 14M.
- 2 If MAXSTOR is set to Y, then set STORLIM to 28M.

Specifying the size required

The REGION parameter of the EXEC statement should contain OM to allow Session Manager to manage storage efficiently:

```
// EXEC PGM=ISZSMGR,REGION=OM
```

Handling upgrades and Functional Enhancement PTFs

Supplied configuration samples

From Session Manager release 2.2.00 configuration data is supplied in ISZCON01 for Classic (in library .SISZCONF), and in ISZCON02 for OLA (in library .SISZSCNF).

Most OLA dataset names have been changed so that all low level qualifiers start with '.SISZ' and several have been added. Customer OLA datasets start with '.SISZC' and are supplied empty. Base samples datasets start with '.SISZB' and contain members that can be used for new installations, to easily construct and run a basic system quickly and also as examples for more complex configuration. System datasets start with '.SISZS' and contain members that must be present in the executing system. Some system members, for example, MENU, can be tailored by the customer.

From release 2.2.00 all messages are prefixed with ISZM (for example, ISZM1234), all classes are prefixed with ISZCLS (for example, ISZCLSIM) and all commands are prefixed with ISZ or ISZZ (for example, ISZUPDT and ISZZINIT).

The table below shows the pre-2.2.00 sample member names and the ISZ-prefixed names used from release 2.2.00.

Note also that member COMMON has been renamed as ISZCOMON.

See also 'Product changes – Session Manager 2.2.00 and higher' on page 58.

List of samples

Overleaf is a table that lists configuration samples which can optionally be used within a Session Manager installation. They are sorted by member name within OLA DDName.

Detailed information on these configuration statements can be found in the *Technical Reference* manual. The list indicates where these samples can be found in a Classic system and an OLA system, and whether the sample has been updated in the specified release.

All samples are in the supplied Base .SISZCONF dataset (see 'Base member name' in table); and/or for OLA relevant members can be found in datasets with low level qualifiers that start with '.SISZB' or '.SISZS'.

Key to table

- YES = changed, NO = not changed, NEW = new member.
- Numbers in brackets identify the update (listed at end of table) that resulted in the revised/new member.

Rel. 2.2.00 name	Pre-Rel. 2.2.00 name	Base member name	OLA DDName	2.1.00	2.1.05	2.2.00	2.2.05	3.1.00
ADMIN	ADMIN	ISZC21VM		NO	NO	NO	NO	NO
ADMIN2	ADMIN2	ISZC21VM		NO	NO	NO	NO	NO
ISZABCST	BRDCAST	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZACICA	CICSA	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZACICB	CICSB	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZAHDSK	HELPDESK	ISZCAPPL	APPL	NEW (2)	NO	NO	NO	NO
ISZAIMS	IMS	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZAMGR	ISZSMGR	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZAOLA	OLA		APPL	NO	NO	NO	NO	NO
ISZASMM	SMM	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZASMSG	SMSG	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZASYSP	SYSPMENU	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZATCP1	TCP_1	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZATSO	TSO	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZATSOA	TSOA	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZATSOB	TSOB	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZATSOD	TSOD	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZAUPHR		ISZCAPPL	APPL					NEW (11)
ISZAUPAS	UPDPASS	ISZCAPPL	APPL	NEW (1)	NO	NO	NO	NO
ISZAUSER	USER	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZAVTM	VTM	ISZCAPPL	APPL	NO	NO	NO	NO	NO
ISZCLSAD	AD		CLASS	NO	NO	NO	NO	NO
ISZCLSBT	BT		CLASS	NO	NO	NO	NO	NO
ISZCLSIM	IM		CLASS	NO	NO	NO	NO	NO
ISZCLSLA	LA		CLASS	NO	NO	NO	NO	NO
ISZCLSNO	NO		CLASS	NO	NO	NO	NO	NO
ISZCLSSU	SU		CLASS	YES (4)	NO	NO	YES (10)	YES (13)
ISZCLSUS	US		CLASS	NO	NO	NO	NO	NO
ISZCONBT	ISZCONBT		CONFIG	NO	NO	NO	NO	NO
ISZCON01	ISZCON01	ISZCON01		NO	NO	YES (8)	NO	NO
ISZCON02	ISZCON02		CONFIG	NO	NO	YES (8)	NO	NO
ISZECLP			COMMAND					NEW (14)
ISZGALL	ALL	ISZCON01	GROUP	NO	NO	NO	NO	NO
ISZHCFHD	HCFHEAD	ISZCHARD	HCFORMAT	NO	NO	NO	NO	NO

Rel. 2.2.00 name	Pre-Rel. 2.2.00 name	Base member name	OLA DDName	2.1.00	2.1.05	2.2.00	2.2.05	3.1.00
ISZHCFTTR	HCFTRAIL	ISZCHARD	HCFORMAT	NO	NO	NO	NO	NO
ISZHCPA	HCPROFA	ISZCHARD	HCPROF	NO	NO	NO	NO	NO
ISZHCPB	HCPROFB	ISZCHARD	HCPROF	NO	NO	NO	NO	NO
ISZHCRA	HCROUTA	ISZCHARD	HCRROUTE	NO	NO	NO	NO	NO
ISZHCRF	HCROUTF	ISZCHARD	HCRROUTE	NO	NO	NO	NO	NO
EXITPROF	EXITPROF	ISZC21VM		NO	NO	NO	NO	NO
ISZP22DM	ISZE22DM	ISZCPROF	PROFILE	NO	NO	NO	NO	NO
ISZE22SM	ISZE22SM	ISZCPROF	PROFILE	NO	NO	NO	NO	NO
ISZPECLP		ISZCPROF	PROFILE					NEW (15)
ISZPROLA	OLAPROF		PROFILE	NO	NO	NO	NO	NO
ISZPROF	PRODPROF	ISZCPROF	PROFILE	NO	NO	NO	NO	YES (11)
ISZPRO1	PRODPRO1	ISZCPROF	PROFILE	YES (1, 2)	NO	NO	NO	YES (11)
ISZPROV	VTMPROF	ISZCPROF	PROFILE	NO	NO	NO	NO	NO
ISZR0240	R002_40	ISZCAPPL	RANGE	NO	NO	NO	NO	NO
ISZR4150	R041_50	ISZCAPPL	RANGE	NO	NO	NO	NO	NO
ISZSMGR	ISZSMGR		RUSER	NO	NO	NO	NO	NO
ISZSYSBT	SYSTEMBT		SYSTEM	NO	NO	NO	NO	NO
ISZSYSKM	SYSTEMKM		SYSTEM	YES (3)	NO	NO	NO	NO
ISZSYS02	SYSTEM01		SYSTEM	NO	NO	NO	NO	NO
*	*	ISZCTERM		NO	NO	NO	NO	NO
ISZADMN	ISZADMN	ISZC21VM	TERMINAL	NO	NO	NO	NO	NO
ISZADM2	ISZADM2	ISZC21VM	TERMINAL	NO	NO	NO	NO	NO
ISZPCICS	CICS	ISZSCRIP		NO	NO	NO	NO	NO
MENU	MENU	ISZLENPU		NO	NO	YES (5,7)	YES (9)	YES (12)
MENU2	MENU2	ISZLENPU		NO	NO	YES (5,7)	YES (9)	YES (12)
SIGNON	SIGNON	ISZLENPU		NO	NO	NO	NO	NO
SIGNONE	SIGNONE	ISZLENPU		NO	NO	NO	NO	NO
SIGNONP		ISZLENPU						NEW (11)
ISZPUPAS	UPDPASS	ISZLENPU		NEW (1)	NO	NO	NO	NO
ISZBLNKP	ISZBLNKP	ISZPANS				NEW (6)	NO	NO
ISZBLNKS	ISZBLNKS	ISZSCRIP				NEW (6)	NO	NO
ISZTSOS	TSO	ISZSCRIP		NO	NO	NO	NO	NO
	ABOX	ISZCSAMP		NO	NO	NO	NO	NO
ISZSCICS	CICSSTRT	ISZSCRIP		NO	NO	NO	NO	NO

Rel. 2.2.00 name	Pre-Rel. 2.2.00 name	Base member name	OLA DDName	2.1.00	2.1.05	2.2.00	2.2.05	3.1.00
ISZSUPHR		ISZSCRIP						NEW (11)
	DLOGA	ISZCSAMP		NO	NO	NO	NO	NO
ISZSTSOS	TSOSTRT	ISZSCRIP		NO	NO	NO	NO	NO
ISZSTS02	TSO2	ISZSCRIP		NO	NO	NO	NO	NO
	VMLOGON	ISZSCAMP		NO	NO	NO	NO	NO
ISZSYSAD	SYSADMIN	ISZCUSER	USER	NO	NO	NO	NO	NO
ISZUSER	USER*	ISZCUSER	USER	NO	NO	NO	NO	NO

List of updates

(Bracketed numbers in table above)

- 1 Support added for Change Password facility, UPDPASS APPL, UPDPASS APPLID and UPDPASS PANEL.
- 2 Support added for Help Desk facility, HELPDESK APPL and APPLID.
- 3 STANDBY, SYSPLEXTYPE and VERBOSE parameters added.
- 4 Allow the SU-supplied Class to modify user-defined RECOVERYLEVEL settings.
- 5 Change MENU and MENU2 to cater for new 'D' line command to delete a dynamically added session.
- 6 New function to avoid blank screen after reconnect.
- 7 Make MENU and MENU2 give user-defined values for PF1, PF3, PF7, PF8 and PF12 precedence over the SAA values.
- 8 Pre Session Manager 2.2.00 OLA releases specified ISZCON01. From 2.2.00 ISZCON02 is specified.
- 9 New CONCEAL and REVEAL commands.
- 10 Allow the SU-supplied Class to modify user-defined CONCEAL, DROP_SESSION and IDLELOCK settings.
- 11 Provide support for External Security Manager password phrases.
- 12 Provide support for Session Manager submenus.
- 13 Allow the SU-supplied Class to modify user-defined IMSCONVERTC, REJBB, ESMOLAGROUP, DAPPLESMAUTH, ERTIMEOUT and EUTIMEOUT settings.
- 14 Provide support for authorizing a User's access to the system via the Eclipse interface.
- 15 Provide support for a User's access to the system via the Eclipse interface.

General information and instructions

New installations

Classic system

You may optionally use the samples to set up your Classic configuration. When generating additional configurations or updating the supplied samples, it is recommended that you use a different member name and use the Customer dataset for these updates, so that they do not clash with configuration samples supplied in future Session Manager upgrades.

The product is supplied with an empty Classic customer dataset having a low level qualifier of `.SISZCUST`. The supplied samples can be found in the library `.SISZCONF`.

You may wish to use these samples either as supplied (for example, the System Management Menu application) or as a starting point for setting up your own configuration.

Finally, you may want to remove from your configuration any supplied samples that you are not using.

OLA system

Use the Online Administration and/or Batch jobs to further configure your OLA system.

You may wish to use these samples either as supplied (for example, the System Management Menu application) or as a starting point for setting up your own configuration. The supplied samples can be found in the datasets that start with `' .SISZB'` or with `' .SISZS'`.

How you update and rename a sample in your OLA system depends on its type. PANELS and SCRIPTS must be updated by editing the Base configuration member directly after copying the member to your customer dataset, whilst other types can be updated by either the Online Administration Facility or the Batch Facility (see the *Online and Batch Administration* manual for more details).

The product is supplied with several empty OLA customer datasets having low level qualifiers that start with `' .SISZC'`.

When generating additional configurations or updating the supplied samples it is recommended that you use a different member name and use the Customer dataset for these updates so that they do not clash with configuration samples supplied in future Session Manager upgrades.

It is recommended that any new or modified PANELS are saved in library `.SISZCUST` for Classic and `.SISZCCNF` for OLA, so that your modifications are not lost after future upgrades. To activate these definitions, create a new member called for example, USERPANS, containing COPY statements for the new and/or modified Panel members and save it, for OLA users in the `.SISZCCNF` library and for Classic users in `.SISZCUST` library. Then insert a COPY statement for, in our case, USERPANS in each ISZCONxx member. For OLA users, add the COPY USERPANS after the COPY ISZCOMON in the `.SISZCCNF` library. For Classic users, add the COPY USERPANS at the end of the ISZCONxx member in the `.SISZCUST` library.

Finally, you may want to remove from your configuration any supplied samples that you are not using.

Upgrades and Functional Enhancement PTFs

Classic system

Review the list of modified/new sample configuration members on page 89. If sample configuration members that you originally used (either as supplied or modified to create your own configuration) have been updated between your current release and this release or Functional Enhancement PTF, and you require these updates, then you will need to do one of the following in your new Customer Configuration dataset:

either

- Incorporate your local modifications into the newly supplied version of the sample configuration. It is recommended that you use a different name for your members in the Customer datasets so that they do not clash with configuration samples supplied in future Session Manager upgrades.

or

- Compare the configuration members in your current release (these could be either samples as originally supplied or samples that were modified to create your own configuration) with the supplied configuration members in the new release, to identify updates. If these updates are required in your configuration then make the necessary changes to incorporate them.

The product is supplied with an empty Classic customer dataset having a low level qualifier of `.SISZCUST`. The supplied samples can be found in the library `.SISZCONF`.

It is recommended that all customer configuration changes should be saved in the supplied Classic customer dataset with a low level qualifier of `.SISZCUST`.

Finally, you may want to remove from your configuration any supplied samples that you are not using.

OLA system

Review the list of modified/new sample configuration members on page 89. If sample configuration members that you originally used (either as supplied or modified to create your own configuration) have been updated between your current release and this release or Functional Enhancement PTF, and you require these updates, then you will need to do one of the following:

either

- Incorporate your local modifications into the newly supplied version of the sample configuration. It is recommended that you use a different name for your members in the Customer datasets so that they do not clash with configuration samples supplied in future Session Manager upgrades.

or

- Compare the configuration members in your current release (these could be either samples as originally supplied or samples that were modified to create your own configuration) with the supplied configuration members in the new release or Functional Enhancement PTF to identify updates. If these updates are required in your configuration then make the necessary changes to incorporate them.

You may also wish to use some of these samples, either as supplied or as a starting point for setting up further updates to your own configuration. To achieve this, copy the required sample(s) to your OLA Customer datasets. The supplied samples can be found in the datasets that start with ‘.SISZB’ or with ‘.SISZS’.

The product is supplied with several empty datasets having low level qualifiers that start with ‘.SISZC’.

How you update a sample configuration member depends on its type. PANELS and SCRIPTS must be updated by copying and renaming the member to your customer dataset and then editing directly, whilst other types can be updated by either the Online Administration Facility or the Batch Facility (see the *Online and Batch Administration* manual for more details).

When generating additional configuration or updating the supplied samples it is recommended that you use a different name for your members in the Customer datasets so that they do not clash with configuration samples supplied in future Session Manager upgrades.

Finally, you may want to remove from your configuration any supplied samples that you are not using.

For further assistance please contact your local support representative.

Supplied exit samples

Overleaf is a table that lists exit samples which can optionally be used within a Session Manager installation.

References to these exits can be found in ‘Session Manager user exit processing’ on page 197. Also review member ISZCEXIT.

Some Exit samples are in the supplied .SISZCONF library in Source format and some in library .SISZAUTH in executable format (for a list see ‘Installing the exits’ on page 204).

Key to table

- YES = changed, NO = not changed, NEW = new member.
- Numbers in brackets identify the update (listed at end of table) that resulted in the revised/new member.

Member name	2.1.00	2.1.05	2.2.00	2.2.05	3.1.00
ISZATTAB			NEW (8)	NO	NO
ISZEPTKT	NO	NO	YES (5,6)	NO	NO
ISZESAMP	NO	NO	NO	NO	NO
ISZE00DR	YES (1)	NO	NO	NO	NO
ISZE05VR	NO	NO	NO	NO	NO
ISZE08CP	NO	NO	NO	NO	NO
ISZE09DR	NO	NO	NO	NO	NO

Member name	2.1.00	2.1.05	2.2.00	2.2.05	3.1.00
ISZE09MA Also see ISZC09MA	NO	NO	NO	NO	NO
ISZE09VR	NO	NO	NO	NO	NO
ISZE09WN	NO	NO	NO	NO	NO
ISZE21GA	NO	NO	NO	NO	NO
ISZE21SF	YES (1,2)	YES (3)	YES (4)	NO	NO
ISZE21PH					NEW (11)
ISZE21VM Also see ISZC21VM	NO	NO	NO	NO	NO
ISZE22DM	NO	NO	NO	YES (10)	NO
ISZE22VM	NO	NO	NO	NO	NO
ISZE25AT			NEW (7)	NO	NO
ISZE25CM	NO	NO	NO	NO	NO
ISZE26IT				NEW (8)	NO
ISZE31PT	NO	NO	NO	NO	NO
ISZE33AT			NEW (7)	NO	NO
ISZE33NY	NO	NO	NO	NO	NO
ISZE36IS				NEW (9)	NO
ISZE39AT			NEW (7)	NO	NO
ISZE39SM	NO	NO	NO	NO	NO
ISZE79ET	NO	NO	NO	NO	NO
ISZE79NY	NO	NO	NO	NO	NO

List of updates

(Bracketed numbers in table above)

- 1 Support added for Change Password facility.
- 2 Support added for checking that user has access to application ACB name and terminal name.
- 3 Terminal type and name added to messages 4014 and 4015.
- 4 Supply Session Manager generic resource name, if required, when issuing RACROUTE VERIFY.
- 5 Generate Passticket using an alternative *userid*.
- 6 Generate Passticket using an alternative value in place of the VTAM applid.
- 7 Provide user audit trail via SMF records for specific applications.

- 8** IDLEDISC, IDLELOGOFF and IDLELOCK terminal expiry exit.
- 9** SIDLTIME session expiry exit.
- 10** Provided processing of the new DROP_SESSION and DYNMDROPSSESSION parameters.
- 11** Provide support for External Security Manager password phrases; allow multiple PROFILES to be assigned by the E21 exit; and enable ISM signon process to be multi-threaded.

General information and instructions

New installations

Review the exits and if required, modify and install as directed in 'Session Manager user exit processing' on page 197. If you need to update an exit, it is recommended that you rename your exit so that it does not clash with exit names supplied in future Session Manager upgrades.

Upgrades and Functional Enhancement PTFs

Review all the exits that you are currently using against the list above to check if your exits have been updated and if you require these updates. If an exit that you are utilizing (either as supplied or modified to create your own exit) has been updated between your current release and this release or Functional Enhancement PTF, and you require these updates, then you will need to do one of the following:

either

- Incorporate your local modifications into the newly supplied version of the exit. It is recommended that you rename your exits so that they do not clash with exit names supplied in future Session Manager upgrades.

or

- Compare the supplied exit in your current release with the supplied exit in the new release to identify updates. If these updates are required in your exit then make the necessary changes to incorporate them.

For further assistance please contact your local support representative.

Supplied JCL samples

Below is a table that lists JCL samples used to install, upgrade and administer an Session Manager installation. References to these jobs will be found within various Session Manager manuals.

JCL samples are in the supplied .SISZCONF library.

Nearly all the samples will change between releases or Functional Enhancement PTFs as the version number would have been updated, but other modifications could also have been made. It is highly recommended that a new copy of the JCL is taken for any new Session Manager release or Functional Enhancement PTF and modified to your installation standards.

Key to table

- YES = changed, NO = not changed, NEW = new member.
- Numbers in brackets identify the update (listed at end of table) that resulted in the revised/new member.

Member name	2.1.00	2.1.05	2.2.00	2.2.05	3.1.00
ISZATGBD				NEW (6)	YES (9)
ISZBAJOB	YES	NO	YES (4)	YES	YES (7)
ISZBASM0	YES	NO	YES (4)	YES	YES (7)
ISZBASM1	YES	NO	YES (4)	YES	YES (7)
ISZBASM2	YES	NO	YES (4)	YES	YES (7)
ISZBASM3	YES	NO	YES (4)	YES	YES (7)
ISZBASM4	YES	NO	YES (4)	YES	YES (7)
ISZBASM5	YES	NO	YES (4)	YES	YES (7)
ISZBASM6	YES	NO	YES (4)	YES	YES (7)
ISZBASM7			NEW (3)	YES	YES (7)
ISZCCJCL	YES (1)	NO	YES (4)	YES	YES
ISZCCJOB	YES	NO	YES (4)	YES	YES (7)
ISZCOJOB				NEW (5)	YES (7)
ISZCOLA	YES	NO	YES (4)	YES	YES (7,8)
ISZCSTRT	YES (2)	NO	YES (4)	YES	YES (7,8)
ISZEASM	YES	NO	YES (4)	YES	NO
ISZLOJOB	YES	NO	YES (4)	YES	YES (7)
ISZOEJOB	YES (2)	NO	YES (4)	YES	YES (7)
ISZSTJOB	YES	NO	YES (4)	YES	YES (7)
ISZUPESC	YES	NO	YES (4)	YES	YES (7)
ISZVCOPY	YES	NO	YES (4)	YES	YES (7)

List of updates

(Bracketed numbers in table above)

- 1 SISZINST dataset changed to SISZCONF dataset.
- 2 SISZWSOS dataset removed.
- 3 Batch example to delete specific sessions.
- 4 New release 2.2.00 datasets and members.
- 5 New job for upgrades from 2.2.00 and higher.
- 6 Sample JCL to create a base AUDIT GDG name.
- 7 Datasets renamed/re-organized.
- 8 TCPIP datasets removed.
- 9 JCL updated to allow the creation of the base DUMP GDG name.

Implementing security

The signon panel provides an entry point to the Session Manager system and allows for various measures of security to be implemented. The security may take the form of passwords supplied by USER statements, from a VSAM file, or from a proprietary security package (RACF, for example) which is accessed using the Exit facility provided by Session Manager.

You should choose between:

- 1** using passwords specified on the USER statement. This is the least secure method of controlling access to Session Manager.
- 2** validating userids and passwords using VSAM files. This is the best method to use if you do not have a dedicated External Security Manager (ESM).
- 3** using one or more ESMs, such as RACF, to authenticate users and to determine a user's Session Manager capabilities. This is probably the best method if your users are already set up in an ESM.

Options 2 and 3 are implemented using the Session Manager security user exit point. Examples of user exit code are provided with the product.

Option 3 allows the list of applications displayed by Session Manager to each user to be tailorable via ESM security profiles. These are termed 'dynamic menus'.

See 'Defining security and implementing dynamic menus' on page 289 for information on how to define and configure the various security regimes.

Accessing applications

This section provides brief information on setting up your applications so that they can be accessed by a Session Manager. For more detailed information, including examples, please refer to 'Setting up applications' on page 167.

Choosing an ACB

Once a terminal is signed on to Session Manager, up to 9999 different sessions may be activated from that terminal. Session Manager does this by simulating a terminal for each session using a VTAM ACB. Once the session is established, the application behaves as if it is writing to a physical device, but in fact it is communicating with the terminal using the ACB which can be thought of as a logical terminal.

If using VTAM sessions, the PARSESS parameter is coded on the logical terminal VTAM ACB definitions to allow multiple sessions to exist between that logical terminal and an application.

Some applications allow only one simultaneous session per terminal or ACB (logical terminal). This is called *non-parallel session processing*. Examples include CICS and IMS™. If an application can use the same ACB for any number of sessions, this is called *parallel session processing*. Examples include Session Manager and TSO.

Application session ACB selection

Session Manager provides various methods of selecting ACBs. More than one method may be suitable for any application; the method used depends on Installation preference. Since Session Manager cannot distinguish between applications capable of handling parallel sessions and those which cannot, it is the responsibility of the Installation to determine the characteristics of the applications. The table below may be used as a guide to establish the appropriate method to be used. Using the standard RANGE method is safest if the parallel capability of the application is unknown.

Session Manager nodes within a High Level Availability Sysplex environment must not use the same ACBs. It is recommended to share the configuration across nodes in a Sysplex environment but if ACBs are specified on the shared RANGE statements then they must be unique.

Using standard method no. 3 below, one possible set-up to overcome this situation would be to add to the SYSTEM statement:

```
%% let gc_acb_suffix = 'cc'
```

(where cc is unique for each SYSTEM statement) and to change your RANGE statements to include this variable, for example:

```
RANGE CICSH
FROM ISZ%%gc_acb_suffix%%001
TO fff
HEX
```

The ACB variable s_acb.nnn, where nnn is the session detail number, can be updated before the session is established and contains the ACB in use after session start.

Standard methods

1 Parallel session ACB

The session ACB may be specified by the SESACB parameter on the SYSTEM statement – this is the default if other methods are not used.

2 ACB substitution

The ACB parameter may be set on the APPL statement or the session definition in the USER, TERMINAL or PROFILE statements. This may be substituted with variables or characters from the user's terminal name or userid.

From Session Manager version 1.3.05 Virtual Terminal Masking (VTM) is supported, to allow the ACB name for a selected session to be assigned according to the applname/userid combination. See 'The Virtual Terminal Masking facility' on page 189 for details.

3 Standard ACB range

The APPL or session definition may specify a RANGE statement. Each RANGE statement defines a common pool of ACBs from which one ACB is allocated per session.

4 Multiple APPL statements per application

Multiple APPL statements for one application may be defined with different characteristics to share a single RANGE.

5 SHAREAPPL ACB range for multiple applications

The SHAREAPPL parameter may be specified on the SYSTEM statement to identify an APPL definition which specifies a RANGE containing a pool of ACBs from which one ACB is allocated per user.

6 User-level session ACB

The ACB name is a unique ACB name allocated to the user. It may be a specific ACB name or it may be selected from a RANGE containing a pool of ACBs, the range being specified by means of a special APPL definition.

When testing the system (generally not useful to an end user):

7 Direct user input

The ACB name may be directly keyed into the field s_acb.nnn on the Application Selection menu.

When the application has a non-standard ACB selection requirement:

8 Menu TPSL

Panel processing logic, provided by TPSL, may be used to set s_acb.nnn.

9 INITSCRIPT

An INITSCRIPT may be defined to set s_acb. The ACB specified will be taken to refer to the current session, whether or not the variable is subscripted with a session detail number.

If Virtual Terminal Masking (VTM) is used then the supplied INITSCRIPT ISZSVTM must be used, or must be called from the user's INITSCRIPT. See 'The Virtual Terminal Masking facility' on page 189 for details.

10 User Exit Point E31

The user exit point E31 Slave Session Initiation may set the ACB name.

Each time an application session is started, Session Manager selects an ACB using one of the above methods. Here are some examples of the methods applicable to some of the more common applications.

Application	Typical ACB allocation method(s)	Parallel?
CICS	(2), (3) or (6)	No
IMS	(2), (3) or (6)	No
TSO	(2), (3), (4) or (5)	Yes*
NETVIEW	(2), (3), (4) or (5)	Yes*

* These applications issue the VTAM CLSDST PASS macro. If a number of sessions exist on an ACB for these applications, two sessions may issue a CLSDST PASS simultaneously causing a possible session swap. To prevent this, do not allow more than one of these applications to share a RANGE. Do not, for example, let TSO and NetView[®] or TSOA and TSOB share the same range.

Further information

For more detailed information on setting up your ACBs, refer to 'Setting up applications' on page 167.

Defining System, Profile and User settings

Session Manager provides considerable flexibility in the definition of the parameters which affect end-user facilities. This is due to Session Manager catering for a wide variation of requirements for defaults and overrides.

Most parameters and options that affect each Session Manager user may be defined on a number of statements (the PROFILE, USER, TERMINAL and APPL statements), although the APPL statement is only used to specify options affecting a session with an application.

Using profiles

The most effective way to configure your Session Manager Installation is through the use of profiles. Using profiles greatly reduces the amount of configuration effort required for each user, and simplifies any future changes.

Break your users down into sub-groups: Finance, Operations, IT and Management, for instance. Although some users will be members of more than one group – Operations and IT, for example – you do not need to create different groups for users who fall into more than one area.

Profiles determine the characteristics, facilities and applications available these groups of users or terminals. They can also be used to override certain system options for those users and terminals using the profile.

When a user signs on they are presented with the Menu panel, applications and options that are specified by their profile(s).

Note Although some types of user may require a high degree of flexibility, you do not need to set up a distinct profile for each set of applications you want them to be able to access. For more details, see 'Flexible application access' on page 107.

Choosing the configuration statements

In order to achieve the desired configuration with the minimum amount of installation and maintenance effort, we recommend that you use the following statements to specify the relevant parameters:

Use...	...if you want the setting to apply to:
SYSTEM	<p>All the users or applications in the system.</p> <p>In an OLA system there is an ISZSYSxx configuration member (stored in the PDS(E) allocated to the DDNAME of SYSTEM) for each Session Manager instance that shares the configuration, where xx is the unique identifier for the instance. A value of BT is not allowed as this is used by Batch Administration and a value of CM is not allowed as this is the reserved name for the common SYSTEM statement ISZSYSCM.</p> <p>The ISZSYSCM member contains SYSTEM parameters common across all Session Manager instances and the ISZSYSxx member just contains the SYSTEM parameters unique to that Session Manager instance. If generated by the Enabler, the ISZSYSxx member will contain the ACB, LOCALNODE, SESACB, STANDBY, SYSPLEXTYPE, TCP and VERBOSE parameters; however the ISZSYSxx and ISZSYSCM members may subsequently be modified, using either the Online Administration or Batch facility, to reflect the installation's requirements.</p>
APPL	One specific application
PROFILE	A particular sub-group of users, such as Managers or Operators
USER	One specific user
TERMINAL	One specific terminal (only effective if no user signon takes place)

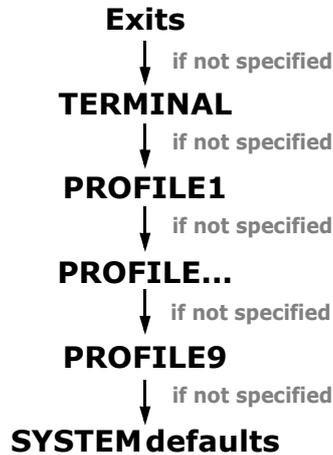
Statements lower in this list will normally override statements higher in the list.

As parameters may sometimes be set to different values in different places, the following diagrams show the logic that is used by Session Manager when deciding which setting to use: this should be taken into consideration when configuring your Installation.

Common enduser parameters

Before signon, or if signon not enabled

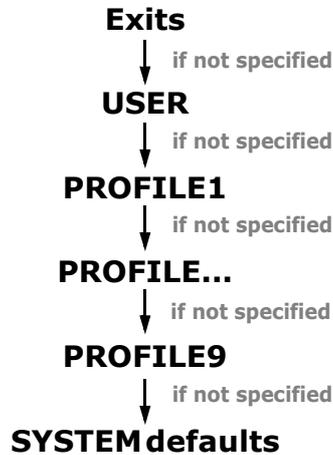
Before a signon has taken place, or if the SYSTEM parameter SIGNON has been set to NO (in which case no signon will take place), the settings to be used will be determined as follows:



For more information on using Exit Points, refer to the *User and Administrator* manual.

After signon

After a signon has taken place the settings to be used will be determined as follows:

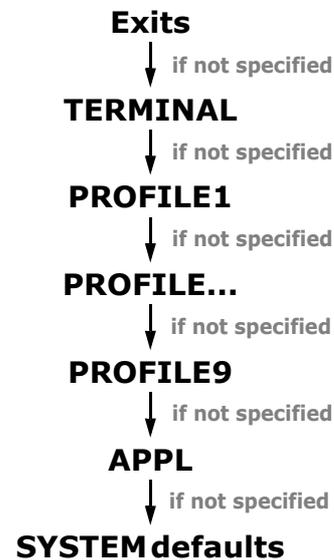


For more information on using Exit Points, refer to the *User and Administrator* manual.

Common session parameters

If signon not enabled

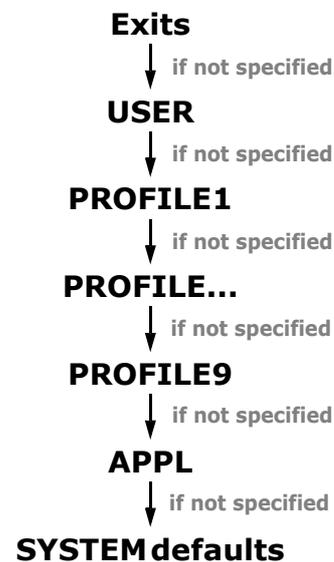
If the SYSTEM parameter SIGNON has been set to NO (in which case no signon will take place), then the settings to be used will be determined as follows:



For more information on using Exit Points, refer to the *User and Administrator* manual.

After signon

After a signon has taken place the settings to be used will be determined as follows:



Determining Profile order

Profiles associated with a user are read in the order they appear on the USER statement, or in the order specified by the ESM and the ESMLEVEL parameter if you have configured your system to use the ESM to determine user Profiles. This is the same order in which they appear when viewed using OLA. The first profile in the list will be Profile1, the second Profile2, and so on.

Setting authorization levels and classes

A user's authorization level determines which commands they can issue, thereby controlling what Session Manager functionality they have access to. Their OLA class determines which configuration items they can access, and what changes they can make within OLA.

The authorization level and OLA class can be set on the SYSTEM, PROFILE, USER or TERMINAL statement, or at signon time by using an ESM. For information on using an ESM, see 'Implementing security' on page 99. For an introduction to OLA, see 'Deciding on a Classic or an OLA system' on page 49.

Different categories of Session Manager command have different default authorization levels:

- Basic and User commands have a default authorization level of 1.
- Operator commands have a default authorization level of 5.
- Administrator commands have a default authorization level of 9.

It is possible to alter the authorization levels of individual commands if the Installation requires it. Please refer to the Commands chapter of the *Technical Reference* for a table showing all the commands in the respective categories and for information on where commands can be issued.

Flexible application access

For some users – System Administrators, for example – it may be desirable to have access to a wide range of different applications. In order to provide this facility without needing to configure individual profiles for many users, application ID overtyping can be used.

On the Session Manager main menu the application ID (APPLID) will normally be displayed, although this may depend on the menu panel being used (see 'Designing panels and menus' on page 111).

To provide temporary access to applications not appearing on the session list, overwrite one of the application IDs on the session list and use the session number, PF key, or selection command relating to the overtyped ID to access the alternative application.

Note The application must have its own APPL statement, and have been set up ready for use. For further information on this, refer to 'Accessing applications' on page 100.

Priority sessions

Another facility to consider, when considering how to limit the amount of user-specific configuration and maximize the use of profiles, is the setting of priority sessions.

This enables users (subject to their OLA class allowing it) to specify certain sessions that they want to appear at the top of their own lists, so that users with different session ordering requirements can use the same profile. For installations using a Classic configuration, priority sessions can be specified by using the `SESPRI` parameter on the `USER` statement.

For information on specifying priority sessions, see the section in the *Online and Batch Administration* manual.

Configuration statement syntax

There are a number of general points of syntax to bear in mind when using configuration statements:

- Several parameters may have any one of four options, Yes (or ON) or No (or OFF).
- Unless otherwise stated, if the parameter is omitted, the default option is No. For example, if `PCTransfer` is not specified on any statement, the setting is `PCTransfer No`.
- If the parameter keyword is specified, but no option is supplied, the default option is Yes. For example, specifying `PCTransfer` with no options, is the same as specifying `PCTransfer Yes`.
- Several parameters define an 'escape' sequence or key, that can be used to invoke a facility, or initiate an action. In all cases this escape must be delimited by a null or a space when issued, and can be any of the following:
 - a character sequence from one to eight characters in length
 - a function key: PF1–PF24
 - an attention key: PA1–PA3, 'ATTN'
 - the 'CLEAR' key (although this should be used with care)
 - the lightpen: 'PEN'
- Where a parameter definition includes a Session Manager statement name, a parameter name, or spaces, it should be enclosed in quotes, or parentheses.

Examples

The following examples illustrate how configuration might be achieved.

Example 1

`USER` statements can be defined as generic and can be applied to a number of user ids. For example, say the Accounts department in a company all have user ids starting `ACCxxxxx`, and they are all to have access to the same set of applications. A `USER` statement can be coded generically as:

```

USER      ACC*
          PROF Profacc
          ...
    
```

This would mean that any user id with 'ACC' as the first three letters would be matched with this USER statement and would use the profile PROFACC. Generically coding USER statements in this way reduces the amount of control statement coding required.

Example 2

Profile PROFA could be coded as:

```

...
KEY PF1 CICSPROD
KEY PF2 User
KEY PF3 User
    
```

Individual USER statements can then be coded to refer to this profile, but if necessary applications can also be defined on the USER statement, for example:

```

USER      Fred                USER      Jill
          PROF  PROFA          PROF  PROFA
          KEY PF2 S09TSO       KEY PF2 S08TSO
          KEY PF3 S02CPGEN     KEY PF3 S08CICS
    
```

Any KEY settings on the USER statement override the 'User' settings on the PROFA profile, so Fred's menu has:

```

PF1 CICSPROD                <-----Taken from PROFILE statement
PF2 S09TSO                   <-----Taken from USER statement
PF3 S02CPGEN                 <-----Taken from USER statement
    
```

Jill's menu has:

```

PF1 CICSPROD                <-----Taken from PROFILE statement
PF2 S08TSO                   <-----Taken from USER statement
PF3 S08CICS                   <-----Taken from USER statement
    
```

Remember that minimizing the amount of configuration performed on USER statements helps to reduce the administrative effort required to set up and maintain Session Manager.

Example 3

Exits can be used to define what applications or commands are available for a user.

E21 signon exit

The exit can use a VSAM file that holds user records defining the sessions available for each user of the system. The exit passes the user id to the file, which then checks there is an entry in the file. If there is an entry, it checks that the password entered is correct, and if it is then the records on the VSAM file for that user are passed to a profile EXITPROF, which is set up as:

```
KEY PF1 User  
KEY PF2 User  
KEY PF3 User  
...  
KEY PFn etc
```

The records passed from the VSAM file then overlay the user's settings and build a set of the applications available to construct a menu. Alternatively, the file can pass just a profile name associated with that user. Refer to 'Implementing security' on page 99 for further information on security considerations.

Designing panels and menus

Menu panel

This panel enables the applications available to a user to be displayed in a set format. The menu also defines how and where messages are displayed, which commands and functions are available to a user, and the escape sequences that can be used. Up to 9999 different sessions may be displayed and controlled from each Menu panel. The Menu panel can be tailored to suit the needs of your Installation; several Menu panels can be created to meet different needs within your organization.

The default menu used by the system is defined by the DEFMENU parameter of the SYSTEM statement. If the parameter is not specified, a menu called 'MENU' is required. The menu to be used can be specified using the common enduser MENU parameter on the SYSTEM, PROFILE, USER or TERMINAL statements.

Sample menu panels

Two sample menu panels can be found in member ISZLENPU (in library .SISZCONF) which may be referenced using COPY ISZLEN for Classic and COPY ISZLEN2 for OLA. MENU is the sample standard menu definition and MENU2 is the sample menu definition for privileged users.

ISZLENPU contains contains the sample standard menu panel definition and the sample Menu definition for privileged users. The panel definitions can be copied and amended so that the format of the menu display conforms to your Installation standards. Several different menu panel definitions can be created to cater for different needs.

It is recommended that any modifications are saved in library .SISZCUST for Classic and .SISZCCNF for OLA, so that your modifications are not lost after future upgrades. To activate these definitions, create a new member called for example, USERPANS, containing COPY statements for the new and/or modified Panel members and save it, for OLA users in the .SISZCCNF library and for Classic users in .SISZCUST library. Then insert a COPY statement for, in our case, USERPANS in each ISZCONxx member. For OLA users, add the COPY USERPANS after the COPY ISZCOMON in the .SISZCCNF library. For Classic users, add the COPY USERPANS at the end of the ISZCONxx member in the .SISZCUST library.

The number of sessions available to each user is held in session variable smax. The variable s_s can be used to determine a subscript value to find variables values for a particular session.

For example, for the profile FRED:

```
PROFILE1 FRED
      KEY PF1      APPLID S05TSO
      SESSION 3    APPLID CICSPGEN
```

For the first session (KEY PF1), variable s_applid.1 will contain S05TSO.

For the second session (SESSION 3), variable s_applid.2 will contain CICSPGEN.

Full descriptions of the SYSTEM statement and DEFMENU parameter can be found in the *Technical Reference*; details of the PANEL statement and related parameters can be found in the *Panel, Script and Variables* manual.

Panel sections

Panels are defined using the PANEL statement and, optionally, the Panel and Script Language (TPSL). Detailed information about both of these can be found in the *Panels, Scripts and Variables* manual.

Session Manager panel definitions consist of the following optional sections:

- Define
- Header
- Content
- Trailer
- Process

The Define section, if specified, must appear first. It is used to define options which apply to the whole panel, such as the width of the panel (80 or 132 characters), whether normal or alternate screen size is to be used, the name of the field that the cursor is to be positioned under, and the maximum number of times that any TPSL loop defined for the panel may be executed.

Each panel display may be optionally divided into a maximum of three sections. These are:

- The Header section is a fixed portion at the top of the display.
- The Trailer section is a fixed portion at the bottom of the display.
- The Content section appears between the Header and Trailer sections. It may have a greater number of lines than the lines available between the last line of the header, and the first line of the trailer. The Content section can be moved backward and forward using the Session Manager scrolling commands.

As stated above, the Header and Trailer sections are fixed in position. It is generally advisable to define at least one of these for each panel, to ensure that the command area is defined in a fixed portion. The cursor should also be positioned initially on a field in a fixed portion, since if it is defined in the Content section, it may not appear on the display where intended when the number of content lines is greater than the lines available.

It is not necessary to define a Content section; a panel definition may consist of just a Header and/or a Trailer section. For example, a Help panel definition may consist solely of a Header section.

The Process section of a panel definition contains TPSL statements which are executed when input is received. This is used to check which attention key or PF key was pressed, or to inspect the command that was entered. This can then be used to validate or control the command that is passed to Session Manager. For example:

```
PROCESS
  If t_command = 'grab' or 'acq' or 'flip'
    Let t_command = 'transfer'
  End
```

Thus, synonyms for the TRANSFER command may be entered on this panel. This can prove particularly useful in installations where users prefer to enter commands in their own language.

The function FORMATMSG can be useful when creating panels, as it enables Session Manager messages to be issued during TPSL processing. Full details can be found in the *Panels, Scripts and Variables* manual.

Panel sub-definitions

The Header, Trailer, Content and Process sections of a panel can specify a CALL parameter. This enables a sub-definition to be included and allows sub-definitions to be shared by several panels. The sub-definitions are defined by the following statements:

- PHEADER
- PCONTENT
- PTRAILER
- PPROCESS

A sub-definition can only be included in a corresponding definition, for example, a PHEADER can only be called from a HEADER or another PHEADER.

Basic field types

There are three basic field types, input, output and literal. These are all defined by field attribute names, and are described in the following sections.

Input fields

Input fields are variables into which the user can enter data. These fields are recognized by having been assigned an attribute of 'IN'. Examples of input fields are the signon id and password. These would normally be entered by a user at the Signon screen. Input fields should be unprotected and are left justified for both numeric and character variables, with any truncation occurring on the right.

Output fields

Output fields are variables which are intended for output, and to which Session Manager ascribes values. These fields may be either Session Manager supplied variables or may be user-definable variables. Session Manager writes the contents of the variable referred to by the name whenever it detects such a field in a screen definition. These fields are recognized by having been assigned an attribute of 'OUT'. Output fields should be protected and numeric fields are right justified; truncation occurs on the left.

For example, to build a useful Menu screen it is necessary to display – for each application that may be selected – a brief description of the application, any data that is passed to the application, and the escape command sequences.

There are several variables which may be used to build a screen layout. These include the date, the time, the signon userid and the luname of the terminal. Session Manager has many such fields which are fully documented in the *Panels, Scripts and Variables* manual.

Literals

Literals are simply character strings which Session Manager displays on the output screen exactly as they are defined on the input records or field definitions. These would therefore be used mostly for headings and for Help screens, and should be protected fields, so that they may not be altered on the screen display. These fields are recognized by having been assigned an attribute of 'LIT', or they may have been allowed to default to LIT, that is, neither 'OUT' nor 'IN' are specified for the field.

Language Pack information

When creating menus and panels for multiple languages using the LANGUAGE parameter the menus or panels with the same name must be created in the same single dataset member.

Further information

Further information on designing panels, views and menus can be found in the *Panels, Scripts and Variables* manual.

National language support

Session Manager enables you to specify the language in which screen text for product panels, messages, and so on, is displayed at a user's terminal. Language Packs (LPs), which are supplied with Session Manager in library .SISZCONF, and OLA messages (in .SISZMSG), contain all the Panels, Scripts, Text and Messages that have been translated. The default LP is English and is always shipped as part of the product. Additional LPs are created and added to the product as they become available.

Each LP has a unique 2-character language id – for example, EN is the language id for the English LP. Contact your local Support Representative to check on available LPs and their corresponding 2-character language ids.

Adding/ customizing LPs

The method used to add/customize LPs depends on the type of your Session Manager configuration.

Classic configuration

LPs are added to the configuration using high level COPY ISZLyy statements:

- The default LP is defined in the COPY ISZLEN statement in the supplied ISZCON01 member, where EN is the language id for English.
- To load a new LP, add an additional COPY ISZLyy statement, where yy is the 2-character language id for the new language, to your ISZCON01 member.

See also

- 'Upgrading Classic to Classic' on page 78.
- 'Converting a Classic to an OLA system' on page 445.

OLA configuration

For an OLA configuration:

- LPs are added to the configuration using high level COPY ISZLyy2 statements:
- The default LP is defined in the COPY ISZLEN2 statement in the supplied ISZCON02 member, where EN is the language id for English.
- To load a new LP, add an additional COPY ISZLyy2 statement, where yy is the 2-character language id for the new language, to your ISZCON02 member.

OLA messages for additional languages are also loaded into the OLA messages dataset .SISZMSG by the SMP/E installation process.

See also

- 'Upgrading OLA-enabled to OLA-enabled' on page 81
- 'Converting a Classic to an OLA system' on page 445.

Specifying the language for screen text

The language in which screen text is displayed for a particular user is specified using the parameter `LANGUAGE xx`, where `xx` is the 2-character language id. This parameter can be included on the `USER`, `PROFILE`, `TERMINAL` and `SYSTEM` statements.

- If the user specifies a `LANGUAGE` value, which must be two characters consisting only of letters of the alphabet or numerals, then Session Manager will load the messages and panels defined with that language. For more information, see:
 - ‘Specifying the language for messages’ below
 - ‘Specifying the language for panels’ below
- If the user has not specified a `LANGUAGE` value, or has specified an invalid value for `xx` (that is, there is no corresponding LP) then the default LP is used.

For more information on specifying the `LANGUAGE` parameter, see ‘Common enduser parameters’ in the *Technical Reference*.

Note If you are using a 3270 emulator then ensure that the appropriate Host Code Page is specified.

Additional customer-defined messages and panels

Specifying the language for messages

A language id can be allocated to appropriate Session Manager messages using the `LANGUAGE` parameter of the `MESSAGE` statement. For details, see the *Technical Reference*.

As for the other language id values, this `LANGUAGE` value must be two characters consisting only of letters of the alphabet or numerals.

When creating menus and panels for multiple languages using the `LANGUAGE` parameter, the menus or panels with the same name must be created in the same single dataset member.

Specifying the language for panels

A language id can be allocated to appropriate Session Manager panels using the `LANGUAGE` parameter of `PANEL`, `PHEADER`, `PCONTENT`, `PTRAILER` and `PPROCESS` definitions. For details, see the *Panels, Scripts and Variables* manual.

As for the other language id values, this `LANGUAGE` value must two characters consisting only of letters of the alphabet or numerals.

When creating menus and panels for multiple languages using the `LANGUAGE` parameter, the menus or panels with the same name must be created in the same single dataset member.

Restrictions

If you are using an LP that contains Double Byte Character Sets (DBCS) then, if you try to use the `CUT` and `PASTE` function on any data that contains DBCS, an error message will be issued informing you that this function is not valid on this type of data.

Summary

Session Manager provides these facilities to enable you to specify the language in which screen text is displayed at a user's terminal:

- Language Packs (LPs) contain all the Panels, Scripts, Text and Messages that have been translated.
- The language in which screen text for Session Manager messages and panels is displayed for a particular user is specified using the `LANGUAGE` common enduser parameter.
- Defining additional customer messages and panels:
 - A language id can be allocated to appropriate Session Manager messages using the `LANGUAGE` parameter of the `MESSAGE` statement.
 - A language id can be allocated to appropriate Session Manager panels using the `LANGUAGE` parameter of `PANEL`, `PHEADER`, `PCONTENT`, `PTRAILER` and `PPROCESS` definitions.

Further information

For further information on how to modify customer-defined panels please refer 'Designing panels and menus' on page 111.

Using scripts with applications

This section provides information on the Session Manager Script facility.

Panels and scripts are used extensively in Session Manager.

The Script facility is provided by the SCRIPT statement. Each script has a unique name and defines a set of executable parameters or 'verbs' which can be thought of as a small program. There are numerous parameters that can be used and scripts are in consequence very flexible and can perform many different functions. For example, a script invoked from a Menu screen can perform a logon to a CICS system.

Scripts usually fall into one of the following categories:

- Session scripts
- Exit scripts
- Command scripts
- Window scripts
- Application Builder scripts

Some of the facilities available when using scripts to communicate with applications are:

- Authorized scripts – scripts able to issue product commands with the highest Session Manager user authority
- Data input streams - scripts that define one or more input data streams to be passed to an application
- Script process tracing - provide the ability to step through scripts and view the inputs and outputs
- The Panel and Script Language (TPSL) - allows conditional processing and assignment of values to variables to take place

Session Manager provides a special language, the Panel and Script Language (TPSL) which enables logic to be added to scripts. Scripts can be caused to wait until certain output is received, for example from an application, and continuation of the script can be made conditional. Input from the terminal may be mapped into variables, passed onto an application, or passed to another script. Scripts can also be nested up to any level.

Full details of all of these facilities, plus detailed information on all the possible uses of scripts, can be found in the *Panels, Scripts and Variables* manual.

Session scripts

Session Manager can pass data streams to an application automatically. This can be done without any terminal input from the user. A session script, as its name suggests, relates to a specific application session, or session parameter and can only be invoked when an ENVIRONSCRIPT, STARTSCRIPT, AUTOSCRIP or ENDSCRIP, parameter is defined in the configuration file. There are two other scripts associated with sessions, that run before a session start (INITSCRIPT) and after a session end (TERMSCRIP). These scripts do not communicate with the application.

All session scripts can be specified at SYSTEM, PROFILE, USER, TERMINAL and APPL levels.

Each session script defines either one, or a series of input data streams to be passed to an application. The script comprises control information, such as SBA and cursor information, and the actual text to be passed. Scripts are able to search for specific information in the data stream output by the application, and TPSL statements may then alter the script processing according to what is found.

Session scripts relate to sessions invoked from a Menu screen and are further subdivided into six categories:

Initscripts

An 'initialization' script may be defined to be run before session start and therefore does not communicate with the application. This type of script may be used to alter session options and parameters. All script parameters used to communicate with an application (for example, INPUT, WAITAPPL and so on) are ignored.

The INITSC command enables a suitably authorized user to disable or (re-)enable the running of the session 'initialization' script specified using the common session parameter INITSCRIPT. For details, see the *Technical Reference*.

Environscripts

This enables all processing common to a specific operating environment to be held in just one script.

Startscripts and Autoscripts

Scripts may be input at session start (Startscript), or by the user issuing a special command sequence defined by the AUTOSEQ or SAUTOSEQ parameters of the SYSTEM, USER, PROFILE and TERMINAL statements (Autoscript).

The STARTSC command enables a suitably authorized user to disable or (re-)enable the running of the session start script specified using the common session parameter STARTSCRIPT. For details, see the *Technical Reference*.

Endscripts

This type of script is run when the user issues the END command for an active session. It is also run when the LOGOFF command is entered, if LOGOFF END has been specified for the user and is in effect for the active session. It enables a script to 'clean up' the session before it is terminated.

Termscript

A script may be defined to be run when a session has terminated and therefore does not communicate with the application. This type of script is generally used to reset session options. All script parameters used to communicate with an application, for example, INPUT, WAITAPPL, are ignored.

Exit scripts

An Exit script may be defined to run in place of, or in addition to, most of the Assembler/COBOL exit points of the Session Manager user exit.

For most exit scripts, this is achieved by specifying the exit point name and the option 'S' on the OPTION statement. A script of the form EXITxx must be defined for the relevant exit, where xx is the exit point. For example, if 'E33 S' is specified on the OPTION statement, a script EXIT33 will be called.

In contrast, for the input (E25) and output (E35) 3270 datastream exits, load module names are identified on the SYSTEM statement by the INPUTEXIT and OUTPUTEXIT parameters respectively. An additional, related, SYSTEM statement parameter, EXITWALEN, specifies the size of the work area made available to the exits. For details on these parameters, see the *Technical Reference*.

Full details of Exit scripts can be found in 'Session Manager user exit processing' on page 197.

Command scripts

Command scripts can be defined to run in place of Session Manager-defined commands. They can also be used to specify user-defined script commands. This is enabled by the CMDSCRIPT parameter of the COMMAND statement. If CMDSCRIPT is specified but no *scriptname* is provided, Session Manager attempts to run a script with the same name as the command. If a *scriptname* is provided, it is this script that is run in place of the command. For example:

```
COMMAND logoff CMDSCRIPT Yes
```

means that a script called logoff is run instead of the command being actioned.

```
COMMAND logoff CMDSCRIPT Yes SNAME fred
```

means that the script fred is run instead of the command LOGOFF being actioned.

Any command name can be used in the script; it is not restricted to Session Manager defined commands. Using this facility a user can effectively define extra user-defined commands. For example:

```
COMMAND bill CMDSCRIPT Yes
```

defines a command that runs a script called bill.

Full details of Command scripts, together with script examples, are provided in the *Panels, Scripts and Variables* manual.

Windows scripts

A Windows script relates to the Session Manager Windows feature and is not session-specific. Windows scripts can be invoked from the Menu panel using the WINDOWS command. Depending on the contents of the script, the terminal screen is divided into areas. Each area usually contains its own command input field, so that each area can perform its own function. Typically, several sessions can be active and viewable simultaneously.

A full description of the Session Manager Windows feature and its usage is provided in the 'Windows feature' chapter of the *User and Administrator Guide*. Information on Windows script verbs is provided in the *Panels, Scripts and Variables* manual.

Application builder scripts

Any of the above can also carry out Application Builder functions by use of certain Application Builder verbs. An Application Builder script relates to the Application Builder feature. These scripts are session scripts, but they have special parameters or verbs that enable output from multiple sessions to be selectively combined and presented on a single panel. Additionally, output from one or more sessions may be used as input to any other session(s). This is achieved by running Application Builder scripts on each of the sessions concerned.

In addition to the normal session script processing, an Application Builder script can start and stop a session, run a script on a specified session and halt scripts that are in progress on any session. Details from any part of an application screen can be copied to a variable and data can be sent to, and received from, a script running on another session.

Further details of the Application Builder feature are given in ‘The Application Builder feature’ on page 433. See the *Panels, Scripts and Variables* manual for further details of Application Builder script verbs.

Understanding execution of a script

The AUDITMSG script parameter can be used to send messages to the audit file during script processing. It is useful in understanding what is happening in an existing script, and for debugging a new script. It is also possible, if necessary, to use the TTPSL command for tracing the logic flow through panels and scripts. The AUDITMSG parameter and TTPSL command are described fully in the *Panels, Scripts and Variables* manual.

Session Manager variables

Session Manager variables are also extensively used in panels and scripts. There are numerous different types of supplied variables (see the *Panels, Scripts and Variables* manual), and Installation-specific variables can also be defined.

Further information

A detailed discussion on the use of scripts can be found in the *Panels, Scripts and Variables* manual, along with some script examples. Sample scripts distributed with Session Manager can also be found by consulting the index file shipped with the product.

Summary

The following statements and parameters are required for the Script facility. A detailed explanation of scripts, together with parameter descriptions, usage notes, script examples, and details on using variables in scripts, is given in the *Panels, Scripts and Variables* manual.

Statement or parameter	Description
SCRIPT statement	Defines a script.
INITSCRIPT parameter	<p>Invoked before the application is initiated.</p> <p>Defines the name of the 'initialization' script to be run before the specified session starts. May be entered on the SYSTEM, PROFILE, USER, APPL and TERMINAL statements.</p> <p>The INITSC command enables a suitably authorized user to disable or (re-)enable the running of the session 'initialization' script. For details, see the <i>Technical Reference</i>.</p>
ENVIRONSCRIPT parameter	<p>Invoked when the application has been contacted.</p> <p>Specifies the name of the script to be run when the specified session starts. May be entered on the SYSTEM, PROFILE, USER, APPL and TERMINAL statements.</p>
STARTSCRIPT parameter	<p>Invoked after the ENVIRONSCRIPT has run.</p> <p>Specifies the name of the script to be run when the specified session starts and any Environscript has run. May be entered on the SYSTEM, PROFILE, USER, APPL and TERMINAL statements.</p> <p>The STARTSC command enables a suitably authorized user to disable or (re-)enable the running of the session start script. For details, see the <i>Technical Reference</i>.</p>
AUTOSCRIPt parameter	<p>Invoked by a user entering a key sequence.</p> <p>Specifies the name of the script to be run if the special command sequence is entered, as defined by the AUTOSEQ or SAUTOSEQ parameter. May be entered on the SYSTEM, PROFILE, USER, APPL and TERMINAL statements.</p>
ENDSCRIPT parameter	<p>Invoked to terminate the application session.</p> <p>Specifies the name of the script to be run if the END command is issued for the session. May be entered on the SYSTEM, PROFILE, USER, APPL and TERMINAL statements.</p>

Statement or parameter	Description (continued)
TERMScript parameter	<p>Invoked after application session has terminated.</p> <p>Specifies the name of the script to be run after the specified session has ended. May be entered on the SYSTEM, PROFILE, USER, APPL and TERMINAL statements.</p>
EXIT parameter	<p>For all but the input (E25) and output (E35) 3270 datastream exits, this parameter is used on the OPTION statement to specify whether a script is to be run in place of, or in addition to, the Assembler/COBOL Session Manager User exit point.</p>
INPUTEXIT parameter OUTPUTEXIT parameter EXITWALEN parameter	<p>Used on the SYSTEM statement, the INPUTEXIT and OUTPUTEXIT parameters identify the load module names of the input (E25) and output (E35) 3270 datastream exits respectively. The related parameter, EXITWALEN, specifies the size of the work area made available to the exits. For details on these parameters, see the <i>Technical Reference</i>.</p>
CMDSCRIPT parameter	<p>Used on the COMMAND statement to specify if a script is to be run instead of Session Manager logic to action the specified command.</p>
WINDSCRIPT parameter	<p>Specifies the name of the script to be run when the WINDOWS command is entered (if the Windows feature is enabled). May be entered on the SYSTEM, PROFILE, USER and TERMINAL statements.</p>

User Exits and the Multiple Exit Driver

User Exits allow a wide range of site-specific operations to take place at a variety of points throughout Session Manager, including, among others, initialization of Session Manager, user signon, user disconnect or signoff, and Session Manager termination.

User Exits can be written into a single program module or into multiple program modules controlled by the supplied Multiple Exit Driver. Most User Exits can also be written as scripts using Session Manager's TPSL language.

See 'Session Manager user exit processing' on page 197 for more details on writing and configuring User Exits.

Setting TCP/IP support (if required)

To start the Session Manager TCP/IP manager, the SYSTEM statement TCP parameter should be included in the configuration file. This can be either at startup, or by using the UPDATE facility in a Classic system, or the PUPDATE or by using OLA in an OLA system. The TCP/IP manager is started at the end of the Session Manager startup process or when the UPDATE has completed.

```
[ TCP [Yes | No | ON | OFF]
      [ STN3270 port-no ]
      [ [ IBM] [ IUCVname resource-name ]
```

A full description of the TCP parameters can be found in the SYSTEM statement section of the *Technical Reference* manual.

Enabling Eclipse support

To use the new Session Manager Plug-in, configure Session Manager with Eclipse support.

Mandatory steps

You must:

- 1 Ensure that TCP=YES is specified on the SYSTEM statement.
- 2 Add the parameter ECLIPSESERVER *port_no* to the SYSTEM statement, where *port_no* specifies the port on which the Eclipse server is to listen for connection requests. The *port_no* specified must be used for Eclipse only.
- 3 Ensure that the LOCALNODE parameter on the SYSTEM statement has a value specified.
- 4 Ensure that *all* users of the Eclipse feature have a USER statement specified.

Optional steps

You can also:

- 1 Add the parameter RCMDTIMEOUT *timeout_value*, if required, to specify a time in minutes after which a command sent to a remote system will time out.
- 2 Specify values for the common enduser parameters ERTIMEOUT to specify a timeout value in seconds that the Eclipse server will wait for a response to a request before issuing an error message and EUTIMEOUT to specify a value for the number of minutes of inactivity after which an Eclipse user is to be logged off.
- 3 Authority to access the system via the Eclipse interface is determined by a combination of the User's AUTHORITY, and the AUTHORITY of the supplied command definition, ISZECLP. If the User's AUTH is less than that defined for the ISZECLP command, access to the system is denied, and message ISZ4297E appears in the Eclipse logon dialog box. Command ISZECLP has a supplied AUTH of 9. This should be modified by the installation as required.

A full description of the TCP, ECLIPSESERVER and RCMDTIMEOUT parameters can be found in the SYSTEM statement section of the *Technical Reference* manual. A full description of the ERTIMEOUT and EUTIMEOUT parameters can be found in the chapter 'Common parameters' in the *Technical Reference* manual. See the *User and Administrator* guide and the online help for further details of the Eclipse plug-in.

Note A user may be logged on to Session Manager as a 3270 user and via Eclipse concurrently. Users defined with either SHARESESS or SHAREDISC can not log into the Eclipse Plug-in.

Networking and Sysplex considerations (if required)

If you do not intend to use Sysplex networking or VTAM networking then you can ignore this step.

If you intend to install two or more Session Manager nodes in a Sysplex these additional tasks have to be performed:

- Define a Sysplex User List Coupling Facility structure
- Set up VTAM application session recovery
- Define a Sysplex Group audit log structure
- Define the audit log structure to the MVS Logger.

Sysplex networking itself requires no additional setup tasks other than defining a 1-4 character Sysplex group name using the `SYSPLXGROUP` parameter on each node's `SYSTEM` statement. For details, see 'Sysplex networking and structures' below.

Note Although it is recommended that you perform these tasks, they are not required if you decide not to use Automatic reconnection within a Sysplex and/or the Sysplex Group audit log.

For more information, see the chapter on 'Parallel Sysplex support' in the *User and Administrator* manual.

If you intend to install two or more Session Manager nodes without using Session Manager Sysplex networking then the nodes can be linked using Session Manager VTAM LU0 `LINK` statements. For details, see 'VTAM implications' on page 139.

See also:

- 'Using multiple Session Manager instances – networking' on page 137.

Sysplex networking and structures

To configure a Coupling Facility Resource Management (CFRM) policy, which contains the required components to enable Session Manager to run in a Sysplex, the user needs to run the `IXCMIAPU` utility to define/alter aspects of that policy.

This consists of three steps:

- 1 Defining and activating the Coupling Facility structures
- 2 Defining the audit log structure to the MVS Logger
- 3 Defining the audit `LOGSTREAM` to the MVS Logger.

Defining and activating the Coupling Facility structures

Your installation will have an existing job to define your CFRM policy and we will assume the policy is completely refreshed in its entirety, each time. If your policy is updated, rather than refreshed, then make the appropriate changes. The IBM `CFSIZER` web site for IBM Session Manager can be used as a guide to calculating the sizes of the structures.

Note When sizing the User List CF structure, it may be useful to refer to 'Structure connection specifications' on page 461.

Audit log CF structure

We recommend Duplex operation for the Audit log CF structure. Duplexing provides a mechanism whereby data written to the Sysplex log Coupling Facility is additionally written to a staging dataset. This ensures that data records are not lost in the event of CF failure. The environment to enable this can be implemented when the log stream, and CF structure it resides in, are defined via parameters input to the IXCMIAPU utility.

User List CF structure

We also recommend Duplex operation for the User List CF structure; and for more information on this and other aspects of running a Coupling Facility, see *z/OS MVS setting Up a Sysplex SA22-7625*.

The automatic user reconnection facility relies on the data in the User List CF structure and therefore if this list becomes unavailable and you have not enabled System Managed Duplexing, then a user who reconnects/signs on may be logged on again as a new user, creating a second, duplicate user within the Session Manager Sysplex environment.

Provided the Coupling Facilities are enabled for System Managed rebuild and System Managed Duplex rebuild; that is, they have the following defined:

```
ITEM NAME(SMREBLD) NUMBER(1)
ITEM NAME(SMDUPLEX) NUMBER(1)
```

then the program will support System Managed Rebuilds and System Managed Duplex Rebuilds for the User List CF structure via the z/OS system command:

```
SETXCF START,REBUILD.....
```

and supports the altering of the structure size via the z/OS system command:

```
SETXCF START,ALTER
```

This sample job assumes:

- Your policy name is POLICY1
- You have two Coupling Facilities named CFCC1 and CFCC2.
- Your SYSPLEXGROUP name is PLXA
- Your audit log CF structure name is PLXLOGSTRUCT.

The IBM CFSIZER web site for IBM Session Manager gives:

- a** an initial size value for the user list structure of 2048
- b** a size value for the user list structure of 15000
- c** an initial size value for the audit structure of 6000
- d** a size value for the audit structure of 10000.

Once your job has defined the structures successfully, issue a z/OS command such as the following, to activate the changes:

```

        setxcf start,pol,polname=policy1,type=cfrm

//DEFPL JOB ,.....
//STEP10 EXEC PGM=IXCMIAPU
//SYSPRINT DD SYSOUT=A
//SYSABEND DD SYSOUT=A
//SYSIN DD *
DATA TYPE(CFRM) REPORT(YES)
DEFINE POLICY NAME(POLICY1) REPLACE(YES)
  CF NAME(CFCC1)
  ..... Definitions for your first Coupling Facility
  CF NAME(CFCC2)
  ..... Definitions for your second Coupling Facility

        Your other structure definitions

STRUCTURE NAME(ISMUSER_PLXA) Add CF structure for the user list
  SIZE(15000)
  INITSIZE(2048)
  ALLOWAUTOALT(YES)
  FULLTHRESHOLD(80)
  PREFLIST(CFCC1,CFCC2)
  DUPLEX(ENABLED) Note: Only define if Duplex operation is
                    required and enabled (recommended)

STRUCTURE NAME(PLXLOGSTRUCT) Add CF structure for the Audit log
  SIZE(10000)
  INITSIZE(6000)
  PREFLIST(CFCC1,CFCC2)
  DUPLEX(ENABLED) Enable structure to be Duplexed (recommended)

```

Defining the audit log structure to the MVS Logger

Having defined your log structure you then need to define it to the MVS Logger by submitting the following job suitably modified with your values.

This sample job assumes: that your audit log CF structure name is PLXLOGSTRUCT.

```

//DEFPL JOB ,.....
//STEP10 EXEC PGM=IXCMIAPU
//STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  DATA TYPE (LOGR)
  DEFINE STRUCTURE NAME(PLXLOGSTRUCT)
    LOGSNUM(32)
    MAXBUFSIZE(4092)
    AVGBUFSIZE(512)
    STG DUPLEX(YES) Log stream will be Duplexed (recommended)

```

LOGSNUM

Specifies the number of log streams that can be allocated to the coupling facility structure being defined. LOGSNUM must be between 0 and 512.

MAXBUFSIZE

Specifies the size, in bytes, of the largest block that can be written to log streams allocated to the coupling facility. The value for MAXBUFSIZE must be between 1 and 65532 bytes.

AVGBUFSIZE

Specifies the average size, in bytes, of log blocks written to log streams allocated to the coupling facility.

Defining the SYSPLEXGROUP audit LOGSTREAMNAME to the MVS Logger

Having defined your log structure to the MVS Logger you must then define the SYSPLEXGROUP LOGSTREAMNAME to the MVS Logger by submitting the following job suitably modified with your values.

This sample job assumes that:

- Your audit log CF structure name is PLXLOGSTRUCT
- Your audit log LOGSTREAMNAME is to be called LOG.ALLRECS. This name is defined on the LOGSTREAMNAME sub-parameter of the SYSPLEXGROUP parameter of the SYSTEM statement.
- Your LS_DATACLAS is M4PLEX
- Your HLQ is PLEX
- You will retain the audit data for 30 days.

```
//DEFPL JOB ,.....
//STEP10 EXEC PGM=IXCMIAPU
//STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DATA TYPE (LOGR)
DEFINE LOGSTREAM NAME(LOG.ALLRECS)
STRUCTNAME(PLXLOGSTRUCT)
LS_DATACLAS(M4PLEX)
HLQ(PLEX)
LS_SIZE(1024)
LOWOFFLOAD(0)
HIGHOFFLOAD(80)
STG_DUPLEX(YES) Note: only define as YES if Duplex
operation is required and enabled
(recommended)
RETPD(30) Retains logged data for 30 days
AUTODELETE(YES)
```

LS_DATACLAS

Specifies the 1 to 8 byte name of the SMS management class that will be used for log stream offload data set allocation.

LS_SIZE

Specifies the size, in 4K blocks, of the log stream offload DASD data sets for the log stream.

AUTODELETE (YES/NO)

Specifies when system logger can delete log data, specifying YES means system logger deletes log data when the retention period (RETPD) for the data has expired.

RETPD

Specifies the retention period, in days, for data in the log stream. The retention period begins when data is written to the log stream. The value specified for RETPD must be between 0 and 65536.

HIGHOFFLOAD

Specifies the percent value you want to use as the high offload threshold for the coupling facility space allocated for the log stream. When the coupling facility is filled to the high offload threshold point, system logger begins offloading data from the coupling facility to the DASD log stream data sets.

LOWOFFLOAD

Specifies the percent value you want to use as the low offload threshold for the coupling facility for the log stream. The low offload threshold is the point in the coupling facility, in percent value of space consumed, where system logger stops offloading log data to the log stream DASD data sets.

For additional information on these parameters see *z/OS MVS Setting Up a Sysplex SA22-7625*.

Setting up VTAM application session recovery

See 'VTAM application session recovery' on page 398 for a detailed description of this topic. Session recovery is controlled via parameters on the APPL and SYSTEM statements.

- 1 Use the RECOVERYLevel parameter on the APPL statement to define the recovery level (High, Intermediate or None) of an application. This can also be specified on the USER, PROFILE or SYSTEM statement.
- 2 Use the SYSPLEXType parameter of the SYSTEM statement to define whether a given node is a normal Session Manager instance, a controller or a standby controller.
- 3 Use the STANDBY parameter of the SYSTEM statement to specify the target node for a standby instance (or to specify that it is not acting as a standby).
- 4 Sub-parameters of the SYSPLEXGROUP parameter of the SYSTEM statement specify certain characteristics of a controller.
 - a The WAITFORCntltime sub-parameter specifies the time a Session Manager instance will wait before attempting to communicate with the controller (for example, when the controller is not available, due to not being started or because it is transferring or recovering).
 - b The STANDBYTakeTime sub-parameter specifies the time after which a standby controller will wait before it assumes the role of the controller.

Please refer to the *Technical Reference* for more details of these parameters and sub-parameters.

Setting Audit GDG dataset support (if required)

To start the Session Manager Audit GDG dataset support, the SYSTEM statement AUDITGDG parameter and its sub-parameters should be included in the configuration file. This can be either at startup in a Classic system, or by using OLA in an OLA system, where the system will need to be restarted to pick up the AUDITGDG configuration settings. You also need to tailor and run the sample JCL job ISZATGBD (in library .SISZCONF) to define the audit log GDG base name. A full description of the AUDITGDG parameters can be found in the SYSTEM statement section of the *Technical Reference* manual.

During start up the (+1) generation level dataset within the GDG cycle will be allocated, catalogued and opened. If a failure is encountered during this then an error message will be issued and no Audit log records will be written to the GDG. In order to close the current GDG dataset and allocate, catalogue and open the next (+1) generation without re-starting, the SPIN command should be issued. This will take the following format:

```
SPIN AUDITGDG
```

The Audit GDG will be allocated as a variable blocked (VB) dataset which has a record length of 280 and a block size of 3880. If a failure is encountered whilst writing out a block then an error message will be issued and no further writes will be attempted until the next SPIN AUDITGDG command is issued.

Setting Dump GDG Dataset Support (if required)

Taking dumps to GDGs in Session Manager has a number of advantages:

- The dump is taken of a UNIX clone of the Session Manager address space. This is very fast and so has minimal impact on normal Session Manager processing.
- It takes a full dump of the address space, thereby providing the maximum information required to solve problems more quickly.
- It does not fill up the installation's spool space.

To set up support for taking Session Manager dumps to GDG datasets, the SYSTEM statement DUMPGDG parameter and its subparameters should be included in the configuration file. This can be either at startup in a Classic system, or by using OLA in an OLA system, where the DUMPGDG configuration settings will become active immediately. You also need to tailor and run the sample JCL job ISZATGBD (in library .SISZCONF) to define the dump GDG base name. A full description of the DUMPGDG parameters can be found in the SYSTEM statement section of the *Technical Reference* manual.

When an abend occurs, a UNIX call will be made to clone the address space. The cloned child will then allocate the (+1) generation level dataset within the GDG and write a SNAP dump to it. If a failure is encountered during this, then an error message will be issued and a dump will be taken to spool using the normal non-GDG method. Note that, because this non-GDG dump will be taken in a UNIX cloned address space, it will have the UNIX name BPXAS on the spool. The header line of the SNAP dump, however, will contain the name of the parent address space.

When a dump is taken, message 133 will be issued. Because the UNIX call is almost instantaneous, we recommend adding an INFORM group to message 133 so that the necessary people are informed that an ABEND has occurred.

The Dump GDG will be allocated as a variable blocked (VB) dataset with a record length of 125 and a block size of 1632.

Implementing Automatic Restart Manager (if required)

Session Manager can interface with Automatic Restart Manager (ARM) by using the IBM program ARMWRAP (the ARM JCL Wrapper). ARMWRAP enables ARM support to be added to an application that has not implemented ARM, without having to make changes to the application code.

Note ARMWRAP is made available as a USERMOD; it is not included as part of the base z/OS operating system.

To implement ARM, you need to:

- 1 Define and activate an ARM policy.
- 2 Add ARM REGISTER/DEREGISTER steps to your Session Manager startup JCL.
- 3 Set up the security definitions.

Defining an ARM policy

An ARM 'couple' data set is required and an ARM policy must be defined and activated. For instructions on how to customize ARM to your Installations's requirements, see the z/OS manual *MVS Setting Up A Sysplex*.

Modifying the Session Manager startup JCL

To invoke ARM, make these changes to your Session Manager startup JCL:

- 1 Add an ARM REGISTER step *before* the EXEC PGM=ISZSMGR statement:

```
//ARMREG EXEC PGM=ARMWRAP ,
//          PARM=( 'REQUEST=REGISTER,READYBYMSG=N,ELEMENT=elemname' )
```

where *elemname* is your Session Manager startup jobname (for example, ISMJOB). This statement requests that z/OS should manage restarts for Session Manager, which means that if there is a failure then the program will be restarted automatically.

- 2 Add an ARM DEREGISTER step *after* the EXEC PGM=ISZSMGR statement:

```
//ARMDEREG EXEC PGM=ARMWRAP ,
//          PARM=( 'REQUEST=DEREGISTER' )
```

This statement requests that z/OS should no longer manage restarts for Session Manager.

Setting up the security definitions

Users of ARMWRAP need SAF authorization for UPDATE access to FACILITY class resource IXCARM.*elemtype.elemname*.

For example, if *elemtype* is DEFAULT, *elemname* is ISMJOB (your Session Manager startup jobname), and your Installation's external security system is RACF, then a RACF definition like this is required:

```
RDEFINE FACILITY IXCARM.DEFAULT.ISMJOB UACC(NONE)  
PERMIT IXCARM.DEFAULT.ISMJOB CLASS(FACILITY)  
ID(userid) ACCESS(UPDATE)
```

Using multiple Session Manager instances – networking

The Session Manager networking system permits communication between tasks running on two or more Session Manager instances. This section provides an explanation of some of the terms and concepts used when Session Manager is driving a multiple-system network.

There are several reasons for establishing application sessions using a remote instance of Session Manager:

- the session has a lower priority than the main session management processing. For example, some complex script sessions such as Application Builder could be run in another Session Manager, which would execute in a separate address space or partition, having a lower operating system dispatching priority than the main processing;
- the network load will be reduced for applications which run on remote machines. For examples of this, see ‘Reducing network load by using Session Manager networking’ on page 375.

Whilst signed into one instance, a Session Manager operator can use the remote command facility (the SEND command) to issue commands on other network-connected instances.

Requirements

The networking facility of Session Manager requires that:

Either:

The instances are within a Sysplex where they can be connected using Session Manager Sysplex networking which employs the MVS Cross-System Coupling Facility (XCF) to automatically set up virtual LINK statements.

or:

The instances can be connected via VTAM sessions, requiring an SNA route between them that can be defined by explicit LINK statements which are added to the Session Manager configuration.

Nodes

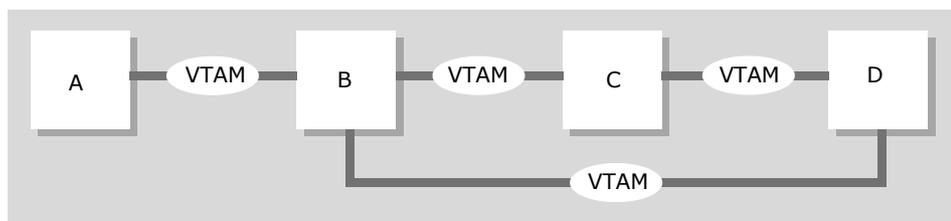
Each instance in the network is referred to as a ‘node’, and is identified by its nodename. Each nodename in the network must be unique and is defined by the LOCALNODE parameter of the SYSTEM statement. Each node on the network has LINK statements which identify a link to another node on the network with which it can communicate.

Note Remember when using Sysplex networking that these LINK statements are virtual and must not be explicitly defined in the configuration file.

Just one type of link may be defined – VTAM LU0, which can be used to connect nodes when there is an SNA route between them.

Routing

Network communication is possible using intermediate nodes as well as adjacent nodes. However, only nodes connected via VTAM LU0 LINKs can be intermediate nodes; Sysplex nodes can not operate as intermediate nodes. In the network diagram below, consisting of nodes connected via VTAM LU0 LINKs, a user on instance A may have a 'REMOTE D' statement in their PROFILE and start a session on instance D. When the session setup request passes through instance B, there is a choice in routing: the shortest route is always used.



Internal names in the network

Each Session Manager instance has an internal name which persists for one execution only. This internal name is referred to in messages as the Network Manager Id and is generally only of interest for diagnostic purposes.

Sysplex networking

Multiple Session Manager instances operating in a Sysplex can use Session Manager Sysplex networking to enable a number of facilities such as:

- Automatic user reconnection.
- Sysplex group wide BROADCAST and MSG commands.
- Sysplex menu provides a single-system view across the Sysplex group.
- Elimination of LINK statements.
- Elimination of RUSER statements.

RUSER statements are not required for nodes within a SYSPLEXGROUP because the user's AUTH and OLAClass, as established on the user's originating node, will be used by the remote node. For nodes in a SYSPLEXGROUP any matching RUSER statements will override the user's AUTH and OLAClass, therefore any redundant RUSER statements should be deleted.

Sysplex networking is activated by specifying the SYSPLEXGROUP parameter on the SYSTEM statement.

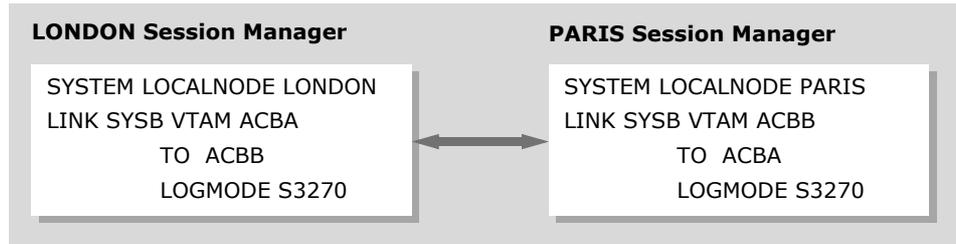
For more information on Sysplex networking see 'Parallel Sysplex support' on page 387.

VTAM implications

For VTAM session support, an ACB must be defined for use by the Link session. This can be done using an APPL definition statement which should have PARSESS=YES defined. Any necessary cross-domain definitions must be in place and activated before Session Manager is able to communicate with remote instances.

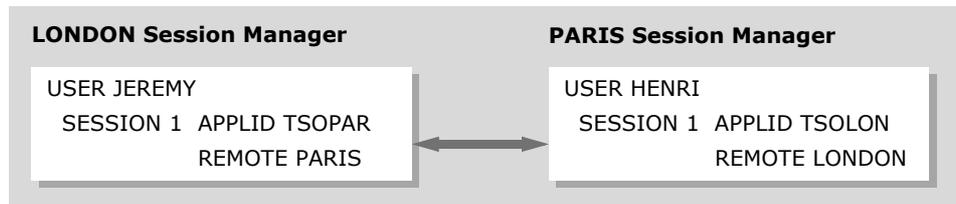
Defining links

The following diagram shows a VTAM LU0 connection example between two Session Manager nodes, LONDON and PARIS:



where ACBA and ACBB are ACBnames used by Session Manager in LONDON and PARIS respectively, solely for the link data traffic. S3270 is a standard logmode which is appropriate for an LU0 session.

This diagram shows a sample user definition, assuming Session Manager in LONDON and PARIS are connected using the VTAM LU0 connection as above:



When user JEREMY, running in LONDON, starts session 1 from the Menu panel, the session is started using the SYSB LINK to PARIS with application TSOPAR which runs in the PARIS Session Manager instance. When user HENRI, running in PARIS, starts session 1 from the Menu panel, the session is started using the SYSA LINK to LONDON with application TSOLON which runs in the LONDON Session Manager instance.

Using Online Administration (OLA)

If you intend to use OLA to administer your Installation then all the Session Manager instances in your Installation should share one set of configuration data. For set-up instructions, see ‘Setting up OLA for single and multiple Session Manager instances’ on page 141.

Using the VTM facility

If you intend to use the Virtual Terminal Masking (VTM) facility then all the Session Manager instances in your Installation should share one set of configuration data. For set-up instructions, see ‘VTM with multiple Session Manager instances’ on page 193.

Further information

For further information on using Session Manager networking, including user affinity, remote commands, and using the REMOTE session parameter and QUERY command, please refer to 'Session Manager networking' on page 373.

Setting up OLA for single and multiple Session Manager instances

Notes

- 1 If you are an existing user of a Classic configuration (that is, all configuration definitions are stored in members of PDS(s) allocated to the DDNAME of CONFIG) then the OLA Enabler performs most of the configuration steps automatically, so please review 'The OLA Enabler' on page 451 before proceeding.
- 2 Usually, OLA will be used to administer a single Session Manager instance, but it can also be used to administer multiple Session Manager instances.

See

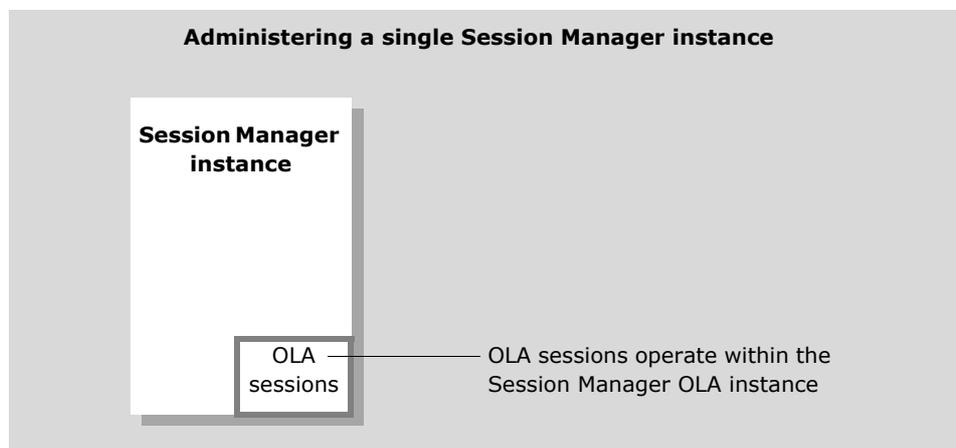
- 'Administering a single Session Manager instance' below.
- 'Multiple Session Manager instances sharing a configuration' on page 142.

Note Full descriptions of the Session Manager configuration statements referred to in these sections can be found in the *Technical Reference*.

Administering a single Session Manager instance

Note OLA can also be used to administer multiple Session Manager instances. For details, see 'Multiple Session Manager instances sharing a configuration' on page 142.

This section describes the setup required to administer a single Session Manager instance, where the OLA sessions operate within the Session Manager instance.



Configuration definitions

If you have performed a basic OLA install, or if the `addola` option (see page 455) is specified when the OLA Enabler is run, definitions such as those shown below will be generated which enable OLA to administer a single Session Manager instance:

- At the PROFILE level, a SESSION definition, where `nnn` is the session number and `admin` is the 'selection command' associated with the session:

```
SESSION nnn admin APPLID iszaola
```

- An APPL definition:

```

APPL iszaola
  INITSCRIPT olasini
  STARTSCRIPT olastart
  INTERNALSESS yes
  DESC "Online Administration"

```

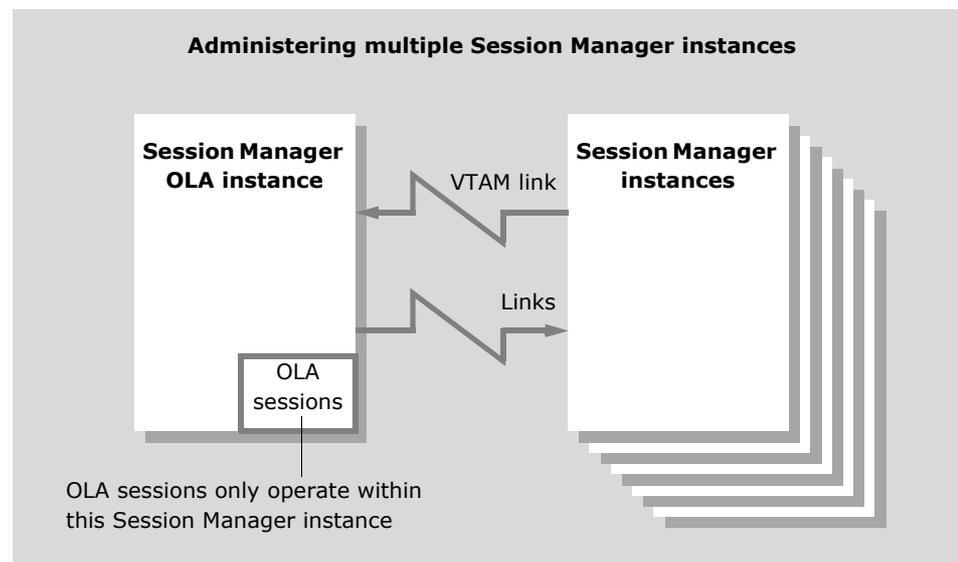
Multiple Session Manager instances sharing a configuration

Note Usually, OLA will be used to administer a single Session Manager instance (see ‘Administering a single Session Manager instance’ on page 141).

This section describes the setup required to administer multiple Session Manager instances. To operate in this way, each instance must share the same configuration datasets.

In a Sysplex environment where the multiple Session Manager instances are assigned to the same SYSPLEXGroup it is mandatory that all the Session Manager instances must share the same configuration datasets. Also any given SYSPLEXGroup must *not* have a mix of Classic and OLA Session Manager instances.

To control contention between users attempting to modify the same item(s), it is recommended that OLA sessions operate within just one Session Manager OLA instance.



Configuration definitions

To enable OLA to administer multiple Session Manager instances, make these changes to the (shared) configuration:

- 1 (Only for a non-Sysplex environment.) Add the VTAM ACB name for the Session Manager OLA instance, the `olastart` STARTSCRIPT and the `DATA` parameter as follows to the APPL definition:

```
APPL iszaola
    INITSRIPT olasini
    STARTSCRIPT olastart
    INTERNALSESS yes
    APPLID xxxxxxxx
    DATA "&t_user&/&t_pass&//iszprola"
    DESC "Online Administration"
```

where xxxxxxxx is the VTAM ACB name.

- 2 (Only for a non-Sysplex environment.) Add this PROFILE definition:

```
PROFILE iszprola
    autoselect 8986
    SESSION 8986
    APPLID iszaola
```

Note This profile is created automatically when you run the OLA Enabler (see page 451). Ensure your users do not use session 8986; and they must not have AUTOSELECT explicitly coded.

Profile ISZPROLA, which is loaded by all Session Manager instances but acted upon only by the Session Manager OLA instance, ensures that the OLA session is AUTOSELECTed for all users signing on to OLA from other instances.

The user's credentials will be passed to the Session Manager OLA instance and, if accepted, the Session Manager signon panel will be bypassed.

- 3 Add appropriate definitions to the ISZCONxx configuration members – see 'ISZCONxx definitions' on page 143.
- 4 (Only for a non-Sysplex environment.) Add LINK definitions for all systems – see 'LINK definitions' on page 145.

The `olasini` script will automatically determine if the environment is Sysplex or non-Sysplex and will assume that the above changes have been implemented for the appropriate environment.

ISZCONxx definitions

There is an ISZCONxx configuration member (stored in the PDS(E) allocated to the DDNAME of CONFIG) for each Session Manager instance that shares the configuration, where xx is the unique identifier for the instance. A value of BT is not allowed as this is used by Batch Administration and a value of CM is not allowed as this is the reserved name for the common SYSTEM statement ISZSYSCM.

Note (Applicable to existing users of a Classic configuration – that is, all configuration definitions are stored in members of PDS(s) allocated to the DDNAME of CONFIG.) ISZCONxx, ISZCONBT (for Batch Administration) and other required members (ISZSYSCM, ISZSYSxx, and so on) will be created if you run the OLA Enabler to implement the OLA-enabled configuration (see page 451).

Each ISZCONxx member, which should be maintained (using the ISPF editor) by the Session Manager implementor, should contain these definitions:

- Any OPTION definition.
- A COPY ISZCINIT statement. (ISZCINIT is the pre-SU maintenance member.)
- An INSTALLSU definition.
- Appropriate AUDITROUTE and TRACEROUTE definitions.
- These configuration logic statements, where *acb_prefix/suffix* is the prefix/suffix for all local ACBs:

```
%% let gc_acb_prefix = 'acb_prefix'
%% let gc_acb_suffix = 'acb_suffix'
```

Specify these variables in RANGE definitions (so that unique ACB ranges are used for each Session Manager instance), LINK definitions (see page 145), and so on. Any configuration parameter that refers to %%gc_acb_prefix%% will use the *acb_prefix* value; similarly, any configuration parameter that refers to %%gc_acb_suffix%% will use the *acb_suffix* value.

Note If you do not modify your RANGE definitions to use %%gc_acb_prefix%% then each Session Manager instance will attempt to use the same range of ACBs and this may fail in some VTAM environments.

- A COPY ISZCOMMON statement.

Member ISZCOMMON, which is processed at Session Manager start up, issues PCOPY and COPY statements to load the rest of the configuration.

Notes

- a OLA recognizes only DDNAMEs that are associated with the Session Manager configuration (see page 448). Therefore, do *not* add PCOPY statements to member ISZCOMMON (or any other member) for DDNAMEs other than those that are associated with the configuration.
- b Variable *t_config_suf* (see *Panels, Scripts and Variables* manual) can be used to refer to the value of xx, the unique identifier for the instance. To copy the ISZSYSxx member, this PCOPY statement should be specified after the PCOPY for the common SYSTEM statement ISZSYSCM:

```
PCOPY SYSTEM ISZSYS%%T_CONFIG_SUF%%
```

ISZSYSxx and ISZSYSCM definitions

There is a ISZSYSxx configuration member (stored in the PDS(E) allocated to the DDNAME of SYSTEM) for each Session Manager instance that shares the configuration, where xx is the unique identifier for the instance. A value of BT is not allowed as this is used by Batch Administration and a value of CM is not allowed as this is the reserved name for the common SYSTEM statement ISZSYSCM.

The ISZSYSCM member contains SYSTEM parameters common across all Session Manager instances and the ISZSYSxx member just contains the SYSTEM parameters unique to that Session Manager instance. If generated by the Enabler, the ISZSYSxx member will contain the ACB, LOCALNODE, SESACB, STANDBY, SYSPLEXTYPE, TCP and VERBOSE parameters; however the ISZSYSxx and ISZSYSCM members may subsequently be modified, using either the Online Administration or Batch facility, to reflect the installation's requirements.

LINK definitions An example will be used to describe how to define LINK definitions to connect Session Manager instances to each other.

Notes

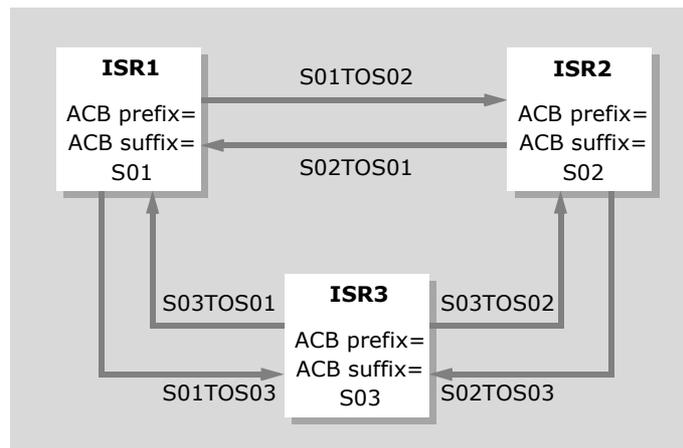
- 1 The LINK definitions are shared/loaded by all Session Manager instances that share the configuration, so the ACBs defined in each LINK definition must use %%gc_acb_prefix%% and %%gc_acb_suffix%%.
- 2 The OLA Enabler (see page 451) creates two LINK definitions – one for the Online Session Manager instance and one for the Batch Session Manager instance. For each additional Session Manager instance, use OLA to add a new LINK definition of the type described in the example below.
- 3 LINK definitions are not required if the multiple Session Manager instances are all running in the same Sysplex Group. Refer to ‘Parallel Sysplex support’ on page 387, and the nolinks parameter on the OLA Enabler (page 451).

Example

Suppose that there are three Session Manager instances, whose LOCALNODE names are ISR1, ISR2 and ISR3. To connect these Session Manager instances to each other, you would need to perform these steps:

- 1 For each instance, define the ACBs to be used for the local ends of the links:

LOCALNODE	gc_acb_prefix	gc_acb_suffix	Local ACBs
ISR1	S01	S01	S01TOS02 S01TOS03
ISR2	S02	S02	S02TOS01 S02TOS03
ISR3	S03	S03	S03TOS01 S03TOS02



- 2 Define a single set of LINK definitions to be used by all of the Session Manager instances, using the LOCALNODE names of the instances for the LINK names:

```
LINK ISR1 VTAM %%gc_acb_prefix%%TOS01 TO
                S01T0%%gc_acb_suffix%% LOGMODE logmode
LINK ISR2 VTAM %%gc_acb_prefix%%TOS02 TO
                S02T0%%gc_acb_suffix%% LOGMODE logmode
LINK ISR3 VTAM %%gc_acb_prefix%%TOS03 TO
                S03T0%%gc_acb_suffix%% LOGMODE logmode
```

Note Prefixing and suffixing `gc_acb_prefix` and `gc_acb_suffix` with two consecutive percent signs (`%%`) causes their values to be substituted *before* the contents of the definition are passed to the Session Manager parser.

Consider the instance having a LOCALNODE of ISR1. For ISR1, `gc_acb_prefix` and `gc_acb_suffix` have the value S01, so these LINK definitions will be used:

```
LINK ISR1 VTAM S01TOS01 TO S01TOS01 LOGMODE logmode
LINK ISR2 VTAM S01TOS02 TO S02TOS01 LOGMODE logmode
LINK ISR3 VTAM S01TOS03 TO S03TOS01 LOGMODE logmode
```

Note A LINK having the same name as the LOCALNODE is *not* activated so, in this case, the LINK named ISR1 would *not* be activated.

Debugging and problem diagnosis facilities

A number of debugging facilities are available in Session Manager which, when used in conjunction with other tools, should enable System Administrators to troubleshoot potential problems with users, sessions and scripts.

If you are unsure what action to take in a particular situation, please contact your local Support Representative before making changes.

User and session problems

If problems are encountered with users or sessions, such as inability to logon, or hanging sessions, Session Manager's TRACE facility, along with the VTAM buffer and I/O traces (as applicable), can help to identify problems. The Session Manager TRACE command enables the analysis of a number of different types of activity.

For further information on the TRACE command, and a detailed discussion of troubleshooting techniques, refer to the *Technical Reference*.

Script and menu problems

A number of different options are available to assist with troubleshooting Session Manager scripts and menus.

If troubleshooting menus, the most straightforward methods are to add literals to scripts (see 'Designing panels and menus' on page 111) so that additional information is displayed when the menu definition is used, or to use the FORMATMSG function, which enables a panel to issue Session Manager messages.

If troubleshooting scripts, the AUDITMSG script parameter can be used to write messages to the audit file at various stages through the script execution.

If these methods are not sufficient, it is possible to use the TTPSL trace command. Before using this, it will be necessary to obtain the task number for the script you wish to trace, by using the QTASK command.

Further information on using literals, the AUDITMSG and FORMATMSG parameters, and the TTPSL command can be found in the *Panels, Scripts and Variables* manual. Further information on the QTASK command can be found in the *Technical Reference*.

Locked keyboards

Methods are available to unlock terminal keyboards when an application fails to respond, or is very slow doing so. Detailed information on the steps you can take can be found in chapter on the ATTN key and locked sessions, in the *Technical Reference*.

Additional commands

Some additional Session Manager problem diagnosis commands are available. These – DLOG, DSTORE, DTERM and DUMP - should be used only as instructed by your local Support Representative.

Further information

For more detailed information, refer to:

- the documentation for the individual commands, and the chapter on problem diagnosis and reporting, in the *Technical Reference*;
- the *Panels, Scripts and Variables* manual for information on working with scripts;
- the chapter on diagnostic material in the *Message and Codes* manual for a comprehensive list of the facilities available, and when to use them.

Access from CICS to Session Manager

Access from CICS to Session Manager is optional. If you wish to access Session Manager from CICS, refer to 'How to set up and use the CICS front-end' on page 337.

CHAPTER 6

Performing a basic configuration

This chapter aims to assist you in getting a simple system up and running once it is installed. It is recommended that any extensive or more complex changes are reserved until you are more familiar with the product.

The subjects covered in this chapter are:

- 'Before you begin' on page 152
- 'Customer, samples and system configuration datasets' on page 155
- 'Control statement summary' on page 159
- 'Basic statements required' on page 162

Before you read this chapter, you should familiarize yourself with 'Post-installation configuration issues' on page 85.

Before you begin

Once IBM Session Manager for z/OS has been successfully installed, and any required changes have been made to VTAM and the operating environment, you can begin to configure the system to meet the needs of your Installation. It is recommended that you have copies of the other Session Manager manuals available before proceeding so that they can be referenced when required.

How configuration is achieved

Session Manager can be tailored to Installation requirements using:

- **Product configuration statements**

If you are not using Online and/or Batch Administration (see below), control statements are read from source members in the PDS(s) allocated to the DDNAME of CONFIG. Full descriptions of each Session Manager configuration statement and each Session Manager command can be found in the *Technical Reference*.

- **Online and Batch Administration**

Online Administration: Instead of supplying product configuration statements directly, OLA enables administrators and end-users of Session Manager to tailor the product using a series of menus, lists and attribute display panels.

Batch Administration: If many changes are required to a large number of configuration definitions, this capability enables administrators and end-users of Session Manager to tailor the product using a batch job.

- **Panels and Scripts**

Installation-specific facilities can be created using Session Manager Panels and Scripts, and conditional logic can be incorporated using the product's Panel and Script Language (TPSL). For details, see the *Panels, Scripts and Variables* manual. To meet particular needs, a user exit is available, containing several exit points and access to certain variables, enabling user code to be executed. For details, see the *User and Administrator* manual.

The sum total of all the control statements contained in the source files or members is generally referred to as the 'configuration'.

The configuration

This section describes the configuration supplied with the product.

For Classic, as supplied, the control statements (which specify the terminals, users and applications in the system) are read from a Classic configuration – that is, all configuration definitions are stored in members of PDS(s) allocated to the DDNAME of CONFIG.

If Online and/or Batch Administration is used to tailor the product, configuration data is stored in several PDS(E)s. In this configuration, each PDS(E) is allocated to a particular DDNAME and *must be maintained exclusively by Online and/or Batch Administration*.

How the configuration is processed

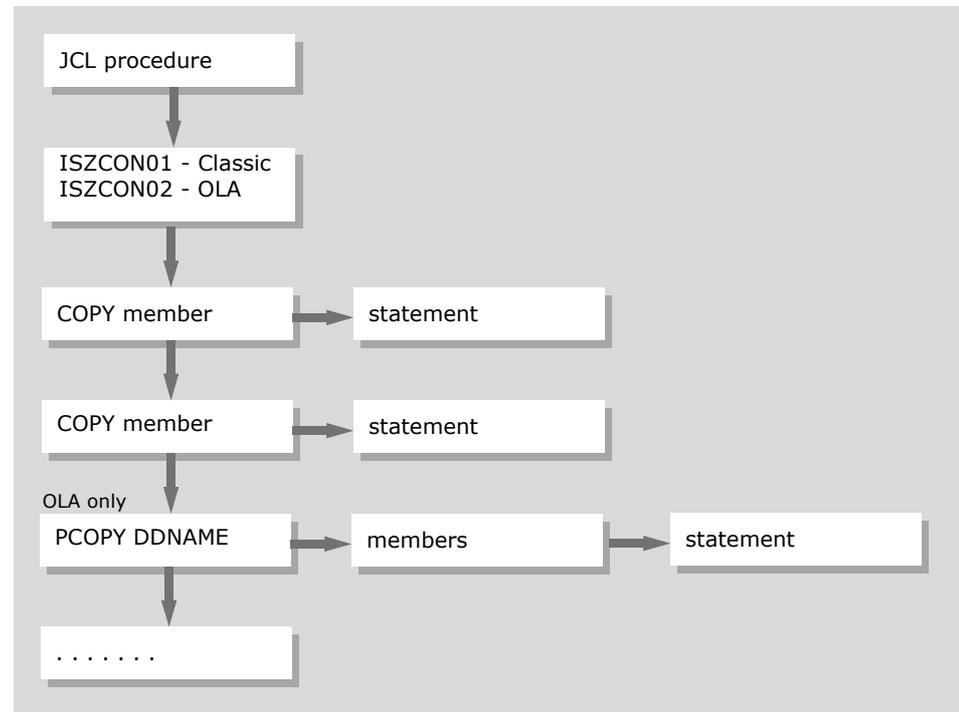
The Session Manager system definition is dynamically loaded from the configuration file(s) at system initiation time and when the UPDATE facility (for Classic) or the PUPDATE facility (for OLA) is invoked, and remains active thereafter. Any dynamic changes made to the Menu screen, such as a change to the escape command sequence, or the updating of a user definable PF key, remain active until Session Manager is shut down, or the user logs off.

The control statements can be modified during processing by, firstly, the Session Manager user exit (at exit point E05) and, secondly, by logic statements within the configuration file itself.

The control statements may be held in several different members and can be included in the main configuration file by means of the COPY statement (or the PCOPY statement in the case of OLA) – see the *Technical Reference* manual for details. The advantage of this method is the reduced time taken to process an in-flight update. If the update affects only one member it can be put into effect using the UPDATE M *member-name* (or PUPDATE *ddname member-name* for OLA). Once Session Manager is active, it supports all sessions for all users signed on to it until either it, or VTAM itself, is shut down.

The first source member processed during configuration is defined in the supplied JCL procedure. By default it is called ISZCON01 for Classic and ISZCON02 for OLA, but the '01' or '02' suffix can be changed if required to suit any Installation standards that may be in force. This is the main configuration member and contains a set of COPY and PCOPY statements. Each COPY statement references another source member, which is then processed sequentially by Session Manager. In addition to containing standard configuration statements, each member processed may in turn also contain additional COPY statements, including further source members. Each PCOPY statement will reference an OLA DDNAME and all members within that DD will be processed.

In this way, Session Manager works through the source members until all statements have been processed. The following diagram illustrates this sequence of operations:



Any naming convention can be used for the members. The members supplied on the distribution media all begin 'ISZ'.

Structure of the configuration

In general, statements may be specified in any order. However, `OPTION` and `INSTALLSU` statements should be placed at the start of the configuration. It is also advisable to place the `SYSTEM` statement next.

Using the supplied configuration

The Session Manager configuration member `ISZCON01` and `ISZCON02` supplied on the Session Manager distribution media contains many examples of the various control statements and parameters required. This can be used as a base from which to work when defining your own system. Note, however, that if you change the name of this member to conform to your Installation naming standards, then you must ensure that this new name is given on the `CONFIG` parameter of the `OPTION` statement.

To change the configuration, you must update the required source members and change the required parameter values, or insert new statements where none have been supplied. The new configuration comes into effect when Session Manager is started, or if the system is already active, when the `UPDATE` or `PUPDATE` command is used.

In addition, a `ISZ£READ` member will have been downloaded during the installation procedure which contains some general information. The `ISZ£INDX` member contains a list of all the source members that are provided with Session Manager. Some of these members are referred to in this chapter, so you may find it helpful to print a copy.

Customer, samples and system configuration datasets

Classic systems

In a Classic system, the configuration is defined in dataset members that are defined to the CONFIG DDNAME. The product is supplied with two CONFIG datasets with low level qualifiers of .SISZCUST and .SISZCONF.

The .SISZCUST dataset is supplied empty and it is recommended that any customer modifications are saved in this dataset. This will make it easier and more straightforward when upgrading to future releases and also will stop future upgrades from overwriting your configuration changes. It is also recommended that the .SISZCONF library is made READ-ONLY.

The .SISZCONF dataset contains all the supplied configuration. After some basic configuration modifications, the supplied configuration can be used to very quickly start a basic system.

The tables in 'Handling upgrades and Functional Enhancement PTFs' on page 89 list the supplied samples within this dataset which you may want to use to tailor in your systems.

Future upgrades

For a Classic system, the system and empty customer dataset will be installed. Then the customer dataset from the previous release will be copied to the new customer dataset.

Also see 'Handling upgrades and Functional Enhancement PTFs' on page 89.

OLA systems

In an OLA system, a specific DDNAME is associated with each statement type, but customer configuration definitions (see page 157), supplied sample configuration definitions (see page 157) and supplied system configuration definitions (see page 157), are held in different datasets. This will make it easier and more straightforward when upgrading to future releases and also will stop future upgrades from overwriting your configuration changes. It is recommended that the system datasets (starting '.SISZS') and the base samples datasets (starting '.SISZB') are made READ-ONLY.

Below is the resulting OLA-enabled configuration – Batch-only statements are shown in **bold**; DISP=SHR parameters have been omitted for clarity:

```
//ACTIVE DD DSN=CUSTOMER.SISZCACT
//APPL DD DSN=CUSTOMER.SISZCAPL
// DD DSN=SAMPLE.SISZBAPL
// DD DSN=SYSTEM.SISZSAPL
//ASCRIP DD DSN=CUSTOMER.SISZCASC
//CLASS DD DSN=CUSTOMER.SISZCCLS
// DD DSN=SYSTEM.SISZSCLS
//COMMAND DD DSN=CUSTOMER.SISZCCMD
// DD DSN=SYSTEM.SISZSCMD
//CONFIG DD DSN=CUSTOMER.SISZCCNF
// DD DSN=SYSTEM.SISZSCNF
// DD DSN=BASE.SYSTEM.SISZCONF
//GROUP DD DSN=CUSTOMER.SISZCGRP
// DD DSN=SAMPLE.SISZBGRP
//HCFORMAT DD DSN=CUSTOMER.SISZCHCF
// DD DSN=SAMPLE.SISZBHCF
//HCPROF DD DSN=CUSTOMER.SISZCHCP
// DD DSN=SAMPLE.SISZBHCP
//HCRUTE DD DSN=CUSTOMER.SISZCHCR
// DD DSN=SAMPLE.SISZBHCR
//LINK DD DSN=CUSTOMER.SISZCLNK
//MASK DD DSN=CUSTOMER.SISZCMSK
// DD DSN=SAMPLE.SISZBMSK
//MESSAGE DD DSN=CUSTOMER.SISZCMSG
// DD DSN=SYSTEM.SISZSMSG
//NODE DD DSN=CUSTOMER.SISZCNOD
//PROFILE DD DSN=CUSTOMER.SISZCPRF
// DD DSN=SAMPLE.SISZBPRF
// DD DSN=SYSTEM.SISZSPRF
//RANGE DD DSN=CUSTOMER.SISZCRNG
// DD DSN=SAMPLE.SISZBRNG
//RUSER DD DSN=CUSTOMER.SISZCRUS
// DD DSN=SAMPLE.SISZBRUS
//SEARCH DD DSN=CUSTOMER.SISZCSRC
//SYSTEM DD DSN=CUSTOMER.SISZCSYS
// DD DSN=SYSTEM.SISZSSYS
//TEMPUSER DD DSN=CUSTOMER.SISZCTMP
//TERMINAL DD DSN=CUSTOMER.SISZCTRM
//TRANSTAB DD DSN=CUSTOMER.SISZCTRN
//UPDATE DD DSN=CUSTOMER.SISZCUPD
// DD DSN=SAMPLE.SISZBUPD
//USCRIP DD DSN=CUSTOMER.SISZCUSC
//USER DD DSN=CUSTOMER.SISZCUSR
// DD DSN=SAMPLE.SISZBUSR
//VTM DD DSN=CUSTOMER.SISZCVTM
```

For an explanation of the difference between customer and system definitions, see:

- ‘What are customer definitions (.SISZCxxx)?’ below
- ‘What are base sample definitions (.SISZBxxx)?’ on page 157

For notes on adding/modifying definitions in this sort of configuration, see:

- ‘Adding/changing/deleting configuration definitions’ on page 158

What are customer definitions (.SISZCxxx)?

In an OLA system, the product is supplied with numerous empty customer datasets. These datasets start with a low level qualifier .SISZC. The exception to this is .SISZCONF, the supplied base configuration, which it is recommended that you make READ-ONLY.

It is recommended that any new or modified PANELS are saved in the appropriate customer dataset. This will make it easier and more straightforward when upgrading to future releases and also will stop future upgrades from overwriting your configuration changes.

Using the supplied JCL, the OLA system will automatically save any modifications made by using the OLA system in the customer datasets.

In an OLA system, customer datasets will contain Installation-specific configuration definitions such as USER, PROFILE and SYSTEM definitions, and new and modified signon panels, scripts, application definitions, and so on.

Of the members associated with the DDNAME of CONFIG, only the definitions in the ISZCONxx members are considered as customer definitions.

A MESSAGE or COMMAND definition that is not identical to the (system) definition that is shipped with Session Manager is considered to be a customer definition.

What are base sample definitions (.SISZBxxx)?

In an OLA system, the product is supplied with numerous samples. These samples are in the datasets with a low level qualifier that starts with .SISZB. Not all DDNAMEs will have a supplied samples dataset. After some basic configuration modifications, the supplied system configuration including these samples can be used to very quickly start a basic system.

These samples can also be used as a starting point when tailoring your systems, however it is recommended that any modifications are saved in the appropriate customer dataset as these sample datasets will be overwritten by any future upgrades.

Using the supplied JCL, the OLA system will automatically save any modifications made by using the OLA system in the customer datasets.

If these samples are not required in your system then they may be deleted. See ‘Adding/changing/deleting configuration definitions’ below

What are system definitions (.SISZSxxx)?

In an OLA system, the product is supplied with numerous system datasets. These datasets start with a low level qualifier .SISZS. It is recommended that you make these datasets READ-ONLY.

For an OLA system, system datasets contain configuration definitions that are supplied with the product, such as MESSAGE and COMMAND definitions, and OLA PANELS and SCRIPTS. Specifically, a system dataset is associated with each of these DDNAMEs:

APPL, CLASS, COMMAND, CONFIG, MESSAGE, PROFILE and SYSTEM.

After some basic configuration modifications, the supplied system configuration including the samples mentioned above can be used to very quickly start a basic system.

These system datasets may contain members that the customer may want to tailor for their own systems, however it is recommended that any modifications are saved in the appropriate customer dataset as these system datasets will be overwritten by any future upgrades.

Using the supplied JCL, the OLA system will automatically save any modifications made by using the OLA system in the customer datasets.

Adding/changing/deleting configuration definitions

For an OLA system, in which customer configuration definitions are split from the sample configuration definitions and the system configuration definitions, any definitions added or modified by OLA or Batch Administration will be saved in the customer datasets.

If a system definition or sample definition is modified then it will be stored in the corresponding customer dataset. Unlike customer definitions, system definitions and sample definitions can not be deleted by OLA. In other words, a definition stored in a system or sample dataset can not be modified or deleted, although it can be replaced by a modified version which is stored in the corresponding customer dataset.

It is recommended that any new or modified PANELS are saved in library .SISZCUST for Classic and .SISZCCNF for OLA, so that your modifications are not lost after future upgrades. To activate these definitions, create a new member called for example, USERPANS, containing COPY statements for the new and/or modified Panel members and save it, for OLA users in the .SISZCCNF library and for Classic users in .SISZCUST library. Then insert a COPY statement for, in our case, USERPANS in each ISZCONxx member. For OLA users, add the COPY USERPANS after the COPY ISZCOMON in the .SISZCCNF library. For Classic users, add the COPY USERPANS at the end of the ISZCONxx member in the .SISZCUST library.

Notes

If you delete a modified system or sample definition (stored in the corresponding customer dataset) then the original system or sample definition will be reinstated. However, the reinstated system or sample definition will not be used until it is activated manually, or Session Manager is stopped and restarted.

To delete the supplied samples, an external utility like TSO/ISPF should be used; or you could comment out the .SISZB datasets in your JCL.

Future upgrades

For an OLA system, the system, samples and empty customer datasets will be installed. Then the customer datasets from the previous release will be copied to the new customer datasets.

Also see 'Handling upgrades and Functional Enhancement PTFs' on page 89.

Control statement summary

Below is a summary of the Session Manager control statements with a brief description of the purpose of each.

- A discussion of the best way to configure the settings for users and applications can be found in ‘Defining System, Profile and User settings’ on page 103.
- A full description of each statement and its parameters can be found in the *Technical Reference* manual.
- The *User and Administrator* manual provides details of which statements and parameters are required for each of the many facilities and features provided by Session Manager.

Statement	Description
OPTION	Specifies some global options such as the user exit that is to be loaded.
LINK	Defines the characteristics of a link to another Session Manager node. A Network node can have as many links defined to other network nodes as necessary. LINK statements are not required when connecting nodes in a Sysplex because virtual LINKs will be automatically created.
COPY	If you do not use Online and/or Batch Administration to tailor the product, this statement enables additional Session Manager control statements to be read from the source library during product initiation.
PCOPY	If Online and/or Batch Administration is used to tailor the product, this statement loads member(s) of the PDS(E)s allocated to a specified DDNAME during product initiation.
SYSTEM	Defines various parameters that are system-wide, that is, they apply to the complete Session Manager environment. See ‘Defining System, Profile and User settings’ on page 103 for details.
PROFILE	Defines one or a group of applications and their related attributes, which may then be assigned to many users or terminals. See ‘Defining System, Profile and User settings’ on page 103 for details.
USER	Specifies attributes to be assigned to specified users of the system. See ‘Defining System, Profile and User settings’ on page 103 for details.
TERMINAL	Specifies the attributes to be assigned to specified terminals regardless of their type. See ‘Defining System, Profile and User settings’ on page 103 for details.
LU	Specifies the attributes to be assigned to specified LU-type terminals.

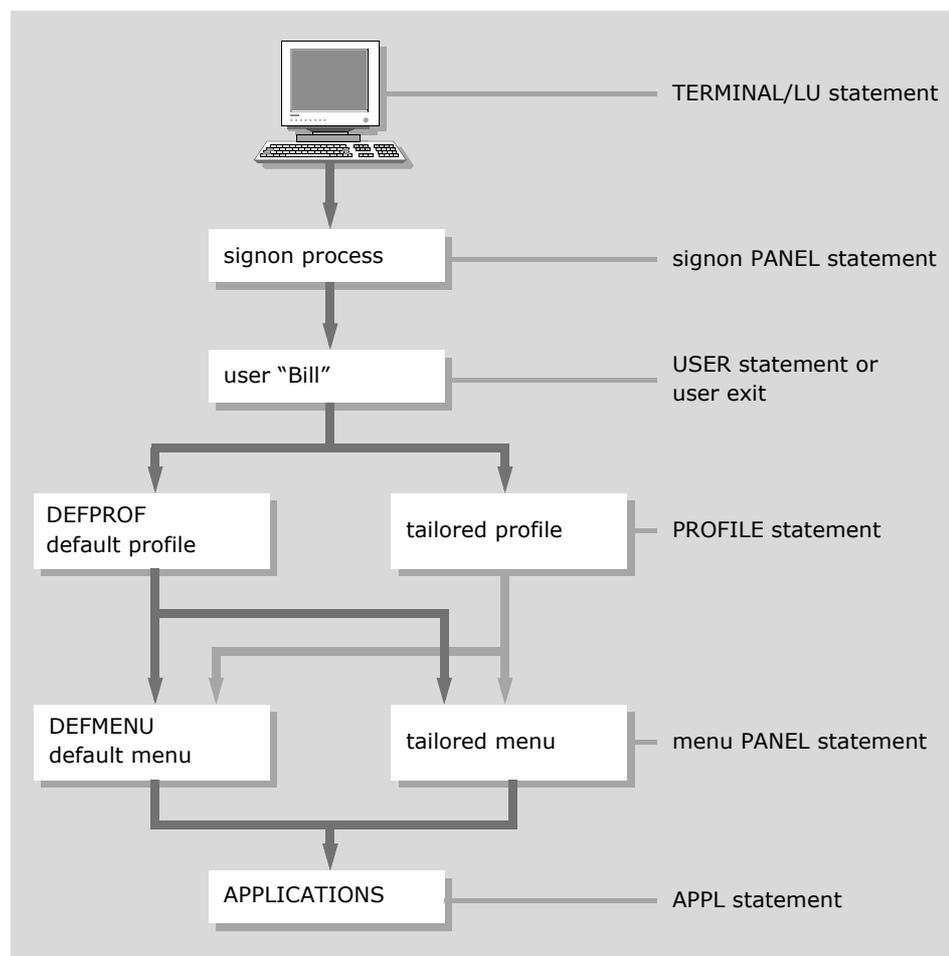
Statement	Description (continued)
APPL	Specifies the characteristics of an application or system. It provides a convenient method of centralizing the definition of application characteristics. See 'Accessing applications' on page 100 and 'Defining System, Profile and User settings' on page 103 for details.
RANGE	Defines ranges of ACB names which can be allocated by Session Manager when initiating sessions with applications which are unable to deal with parallel sessions.
RUSER	Specifies users who can issue remote commands.
GROUP	Defines a named group, which may consist of users, profiles and terminals, for the purpose of receiving messages and broadcasts, and for use with the Spy facility.
HCPROFILE	Defines a set of hardcopy options for use with the screen Hardcopy facility. Full details of the Hardcopy facility are provided in the <i>User and Administrator</i> manual.
HCFORMAT	Defines a set of heading and trailing lines for screen hardcopies.
HROUTE	Defines a hardcopy print route.
COMMAND	Assigns security codes to the Session Manager operator commands, or denotes a script that is to be run when the command is issued.
MESSAGE	Enables the text of Session Manager error messages to be redefined, and enables specific messages to be directed to selected destinations.
AUDITROUTE	Defines the Audit file print route.
TRACEROUTE	Defines the trace print route.
PANEL	Defines screen layouts for the Session Manager system, including the Signon, Menu, Data Display and Help screens. Full details of Panel definition can be found in the <i>Panels, Scripts and Variables</i> manual.
PHEADER	Defines a panel header sub-definition.
PCONTENT	Defines a panel content sub-definition.
PTRAILER	Defines a panel trailer sub-definition.
PPROCESS	Defines a panel process sub-definition.
SCRIPT	Defines automatic input streams. Full details of script definition can be found in the <i>Panels, Scripts and Variables</i> manual.
TRANSTABLE	Defines a table to translate characters in commands, control statements and panels.
INSTALLSU	Ensures that supplied Selectable Units are applied at Session Manager initiation. Full details of Selectable Units can be found in the <i>Technical Reference</i> manual.

Statement	Description (continued)
PATCH	Enables changes to be applied to the executable code. This should only be used at the request of your local support representative.
PATCHSU	Enables changes to be applied to Session Manager Selectable Units. (For a detailed description of Selectable Units, see the <i>Technical Reference</i> .) The PATCHSU statement should only be used at the request of your local support representative.
APPLYSU	Enables a selectable unit to be applied to the Session Manager system after initiation. This should only be used at the request of your local support representative.
REMOVESU	Enables a selectable unit to be removed from the Session Manager system after initiation. This should only be used at the request of your local support representative.

Basic statements required

A user can gain access to applications through the USER, TERMINAL, PROFILE and SYSTEM statement settings, or from rules determined by an ESM (see 'Implementing Dynamic Menus' on page 306.)

Configuration statements, once set up, enable the user to access an application through the following process:



For information on how to use configuration statements most effectively, see 'Defining System, Profile and User settings' on page 103. For information on designing signon and menu panels, see 'Designing the signon panel' on page 290 and 'Designing panels and menus' on page 111.

Supplied statements

In a Classic configuration, all Session Manager configuration control statements (which specify the terminals, users and applications in the system) are read from source members in the PDS(s) allocated to the DDNAME of CONFIG.

In an OLA configuration Online and/or Batch Administration is used to tailor the product, configuration data is stored in several PDS(E)s. In this configuration, each PDS(E) is allocated to a particular DDNAME and *must be maintained exclusively by Online and/or Batch Administration*.

Session Manager comes with supplied definitions that can either be copied, or modified. These include the following:

Default terminal The sample TERMINAL definitions can be found in member ISZCTERM (in library .SISZCONF).

For a discussion on how to use the TERMINAL statement most effectively in your Installation, see ‘Designing panels and menus’ on page 111. A full description of the TERMINAL statement can be found in the *Technical Reference* manual.

Signon panel For a discussion of using signon panels, including the default panels supplied with Session Manager, refer to ‘Designing the signon panel’ on page 290.

Default profile The default profile, DEFPROF, points all first-time users at the default profile. This initial ‘set’ of definitions allows the new user to commence using Session Manager immediately after installation, and to provide a sample base upon which to develop and tailor the system. The sample PROFILE definitions can be found in member ISZCPROF (in library .SISZCONF for Classic, in library .SISZBPRF and .SISZSPRF for OLA).

The default profile used for the system is defined by the DEFPROFILE parameter of the SYSTEM statement. When the OPTION parameter MDPROF is set to M, up to eighteen DEFPROFILES can be specified. If the parameter is not specified, a profile called ‘PROFILE’ is required.

For a discussion on how to use the PROFILE statement most effectively in your Installation, see ‘Designing panels and menus’ on page 111. A full description of the PROFILE statement and DEFPROFILE parameter can be found in the *Technical Reference* manual.

Menu panel For a discussion of using menu panels, including the default panels supplied with Session Manager, refer to ‘Designing panels and menus’ on page 111.

APPL definitions The sample APPL definitions can be found in member ISZCAPPL (in library .SISZCONF for Classic, and in library .SISZBAPL for OLA) can be used to list and define all the applications available at your Installation. These APPL statements can then be referenced from PROFILE, USER and TERMINAL statements to build a list of applications available for individual end-users. For information on how to use these statements most effectively in your Installation, see ‘Designing panels and menus’ on page 111.

Keeping all the application definitions together can save time later if an application definition changes; alterations only have to be coded in one member, rather than several.

A full description of the APPL statement can be found in the *Technical Reference* manual.

Note The applications available to a particular user can also be built dynamically from access rules specified using the Installation’s external security system. See also ‘Implementing Dynamic Menus’ on page 306.

Suggested updates to your system

Classic

For Classic, as supplied, the control statements (which specify the terminals, users, applications and so on, in the system) are read from a Classic configuration – that is, all configuration definitions are stored in members of PDS(E)s allocated to the DDNAME of CONFIG.

For Classic the following members should be updated and referenced in the main configuration file ISZCON01 (supplied in library .SISZCONF):

- ISZCAPPL – see ‘Applications’ below
- ISZCPROF – see ‘Profiles’ below
- ISZLENPU – see ‘Panels’ on page 165
- ISZCUSER – see ‘Users’ on page 166.

It is recommended that any modifications are saved in library .SISZCUST so that you modifications are not lost after future upgrades.

OLA

For Online and/or Batch Administration the control statements (which specify the terminals, users and applications in the system) are stored in several PDS(E)s. In this configuration, each PDS(E) is allocated to a particular DDNAME and *must be maintained exclusively by Online and/or Batch Administration*.

For OLA the supplied control statements are in separate members in their relevant PDS(E)s in libraries that have low level qualifiers that start with ‘.SISZB’ and ‘.SISZS’. With the supplied JCL any modifications will be saved in the customer datasets that start with ‘.SISZC’ and should be updated using the OLA facility:

- APPLs – see ‘Applications’ below
- PROFILEs – see ‘Profiles’ below
- USERs – see ‘Users’ on page 166.

Configuration members

Applications

These control statements should be updated to include all the applications available at your Installation:

Classic	ISZCAPPL (referenced using COPY ISZETAL)
OLA	APPL PDS(E) and RANGE PDS(E)

The samples provide example definitions. It is recommended that you read ‘Accessing applications’ on page 100 in order to understand the options available for application session ACB selection. This will assist in determining how best to define each of your applications.

These control statements also contain sample ranges of ACBs that can be used in application definition. Add new ranges, or amend the samples to suit your Installation. See also ‘Accessing applications’ on page 100.

A full description of the APPL and RANGE statements can be found in the *Technical Reference* manual.

Profiles

These control statements contain several sample PROFILE statements:

```
Classic      ISZCPROF (referenced using COPY ISZETAL)
OLA         PROFILE PDS(E)
```

These can be amended to include applications (as defined in ISZCAPPL, see 'Applications' on page 164) that any user with this profile is to have showing on their Menu screen. Each SESSION or KEY number can reference an APPL statement by coding REFAPPL YES.

If you intend to use OLA (see 'Accessing applications' on page 100) it is advisable to create your own, new, profiles rather than amending the samples provided.

Note When defining the applications that are to be available for users of this profile, you should be aware that if you code SESSION on the PROFILE statement, PF keys cannot be used to access an application or command from the Menu. If you code KEY, then you can either enter the number, or press the PF Key to access the application.

If you want sessions to start automatically, you can specify the AUTOSTART or AUTOSELECT parameters. AUTOSTART can be specified for more than one session. It should be coded for each session as required. AUTOSELECT is specified for one session at PROFILE level and means that the initial Menu display is bypassed and the session is started at signon and automatically selected.

A Menu panel and a PROFILE must be defined before the system can be run. As a first step, these can be the example Menu panel and PROFILE supplied with the product.

A full description of the PROFILE statement and related parameters can be found in the *Technical Reference* manual.

Panels

This member contains sample signon and menu panel definitions:

```
Classic      ISZLENPU (referenced using COPY ISZLEN in library .SISZCONF)
OLA         ISZLENPU (referenced using COPY ISZLEN2 in library .SISZCONF)
```

The panel definition(s) can be amended so that the format of the Menu display conforms to your Installation standards. Several different Menu Panel definitions can be created to cater for different needs.

The sample signon panel can be amended to include your Company logo. In fact several different signon screens can be created to cater for different needs. The default panel is referenced from the SYSTEM statement, but terminal-specific panels can be specified by using the SIGNONPANEL parameter on TERMINAL statements.

For information on designing your own Menu panel, see 'Designing panels and menus' on page 111.

For details of the PANEL statement and related parameters, see the *Panels, Scripts and Variables* manual. See also 'Defining Panels' in the same manual.

The number of sessions available to each user is held in session variable smax. The variable s_s can be used to determine a subscript value to find variables values for a particular session.

For example:

```
PROFILE1 FRED
        KEY PF1      APPLID S05TSO
        SESSION 3    APPLID CICSPGEN
```

For the first session (KEY PF1), variable `s_applid.1` contains S05TSO.

For the second session (SESSION 3), variable `s_applid.2` contains CICSPGEN.

Users

As necessary, add USER statements to these control statements:

```
Classic      ISZCUSER (referenced using COPY ISZETAL)
OLA          USER PDS(E)
```

These statements should refer to the PROFILE and MENU statements as defined in 'Profiles' (see page 165) and 'Panels' (see page 165). USER statements can be specified generically to restrict the amount of coding required.

A discussion on how to use USER, TERMINAL, PROFILE and SYSTEM statements most effectively can be found in 'Defining System, Profile and User settings' on page 103. A full description of the USER statements and its parameters can be found in the *Technical Reference* manual.

The Session Manager User Exit can be used as an alternative to defining USER statements. A full description of the User Exit and using external security routines is provided in the *User and Administrator* manual.

CHAPTER 7

Setting up applications

This chapter provides advice for setting up your applications.

The subjects covered in this chapter are:

- 'Choosing an ACB' on page 168
- 'UNBIND WAIT applications' on page 176
- 'Best fit ACB allocation for RANGE statements' on page 178
- 'Choosing a LOGMODE' on page 184
- 'MTS parameters' on page 186
- 'User-level session ACB support' on page 187
- 'The Virtual Terminal Masking facility' on page 189

Choosing an ACB

An introduction to the factors that need to be considered when choosing an ACB is available in 'Post-installation configuration issues' on page 85. You should read the section about setting up applications in that chapter – which summarizes how to select ACBs – before continuing with the present section, which details those methods.

Method 1: Parallel session ACB

Session Manager uses the SESACB parameter from the SYSTEM statement when an ACB is not specified elsewhere. This is recommended for applications which support parallel sessions such as Session Manager. This method reduces the number of ACBs required in VTAM. However, the application sees all sessions with the same virtual terminal name. Session Manager queries may be used to distinguish which real terminals or users are using that application.

In this example

```
SYSTEM SESACB ISZ001
```

would be specified in Session Manager.

The only requirement in VTAM is that there is a VTAM APPL statement defining the SESACB as an application minor node, for example:

```
ISZ001 APPL ACBNAME=ISZ001 ...
```

If an ACB for an application session is not set by any other method, then any terminal initiating a session with it will use the ACB called ISZ001.

Method 2: ACB substitution

The session should be defined with an ACB subparameter which is available on the SESSION and KEY parameters of the PROFILE, USER or TERMINAL statements. It is also available as a parameter on the APPL statement. An APPL statement ACB setting is overridden by a definition in a PROFILE which is in turn overridden by a definition in a USER or TERMINAL statement.

ACB substitution would generally be used by applications which cannot process parallel sessions, or where the Installation requires that there is an association between the terminal or user and the ACB used for the session.

ACB names may be specified to use the question mark (?) or percent (%) symbols, or variable substitution.

- Use '?' to cause substitution from the user's terminal name. For example:

```
TERMINAL L* PROF PROF1 SIGNON NO
PROFILE PROF1 KEY PF1 APPLID CICSAPPL ACB "T???"
```

When session 1 is started with application 'CICSAPPL', at the terminal with user's terminal name 'L001', the ACB name 'T001' is generated. Similarly ACB 'T121' would be generated for user's terminal name 'L121'.

Although it is not a recognized key in all countries, a hash (#) is allowed in place of a question mark, for backwards compatibility.

Note Since '?' is itself a Session Manager quote character, if it is used in an ACB value then the whole value should be placed within another quote character, for example "AC????".

- Use '%' to cause substitution from the userid. For example:

```
USER L* PROF PROF1
PROFILE PROF1 KEY PF1 APPLID CICSAPPL ACB "T%%"
```

When session 1 is started with application 'CICSAPPL' by user 'LABC', the ACB name 'TABC' is generated. Similarly ACB 'T121' would be generated for userid 'L121'.

- The ACB name may contain any valid variable name. For example:

```
TERMINAL L* PROF PROF1 SIGNON NO
PROFILE PROF1 SESSION 1 CICS APPLID CICSAPPL ACB
'X&t_termid(2,3)'
```

When session 1 is started with application 'CICSAPPL' at the terminal with user's terminal name 'L001', the ACB name 'X001' is generated. Similarly ACB 'X121' would be generated for user's terminal name 'L121'.

Any combination of these types of substitution may be used.

The ACB name generated must not appear in a RANGE statement as this would cause some ACBs from the pool to be reserved outside the RANGE allocation algorithm.

From Session Manager version 1.3.05 Virtual Terminal Masking (VTM) is supported, to allow the ACB name for a selected session to be assigned according to the applname/userid combination. See 'The Virtual Terminal Masking facility' on page 189 for details.

Method 3: Standard ACB range

A unique ACB is allocated per session from a RANGE. This range name is specified on the TERMLOGMODE parameter on the APPL statement. Session Manager searches for an APPL statement if REFAPPL YES is in effect for the session (this is the default). The TERMLOGMODE parameter on the APPL statement must match the logmode name generated for the session. See section 'Application session logmode entry selection' on page 185. An '*' may be specified as the TERMLOGMODE to allow any logmode to match.

A RANGE would generally be used by applications which cannot process parallel sessions and the Installation does not require an association between the terminal or user and the acb used for the session. This may reduce the number of ACBs defined to VTAM.

For example:

```
APPL PRODCICS TERMLOGMODE * ACBR RCICS
APPL TESTCICS TERMLOGMODE * ACBR RCICS
RANGE RCICS FROM T001 TO 100
```

Note Any sessions established with either of these two CICS systems cause an ACB to be allocated from the same range of ACBs, T001 to T100. Session Manager selects the first available ACB, so in this case there is no preset relationship between the CICS terminal ids and the Session Manager user's terminal names.

In VTAM, there must be one APPL statement defined for each ACB in the range of ACBs generated by the RANGE statement as follows:

```
T001 APPL ACBNAME=ACB001,...
T002 APPL ACBNAME=ACB002,...
...
T100 APPL ACBNAME=ACB100,...
```

Note If an application is unable to support parallel sessions, this does not mean the application requires a unique ACB. This means that a range can be shared by two different CICS systems. As in the example, application PRODCICS may have a session using ACB T001 and application TESTCICS may have a session using ACB T001 simultaneously.

In a CICS system using TCT entries, a TCT entry is needed for each ACB:

```
DFHTCT TYPE=TERMINAL,NETNAME=T001,...
DFHTCT TYPE=TERMINAL,NETNAME=T002,...
...
DFHTCT TYPE=TERMINAL,NETNAME=T100,...
```

Specifying TERMLOGMODE * means that any logmode is acceptable. However, if a separate range is required for different types of terminals then code:

```
APPL PRODCICS TERMLOGMODE
D4B32782 ACBR RCICS2
TERMLOGMODE D4B32783 ACBR RCICS3
TERMLOGMODE D4B32784 ACBR RCICS4
TERMLOGMODE D4B32785 ACBR RCICS5
RANGE RCICS2 FROM T001 TO 29
RANGE RCICS3 FROM T030 TO 50
RANGE RCICS4 FROM T051 TO 69
RANGE RCICS5 FROM T070 TO 99
```

Each TERMLOGMODE specifies the bind characteristics required by its associated range of ACBs.

Sessions started with logmodes D4B32782, D4B32783, D4B32784 and D4B32785 use the specified range in the above example. If a session is started with a different logmode to those explicitly coded on the TERMLOGMODE parameters, Session Manager uses a 'best fit' algorithm, described on page 178, to allocate an ACB for the session. The aim is to use the logmode which is most like the logmode which was requested for that session. Refer also to 'Logmode entry selection and REBIND implications' on page 184 and 'Application session logmode entry selection' on page 185.

The same ACB may be defined in more than one range on an APPL statement. Once an ACB is allocated, Session Manager ensures that it is made unavailable in all ranges on that APPL and remains allocated until the session has ended. For example:

```
APPL TESTCICS TERMLOGMODE D4B32782 ACBR RTCIC2
TERMLOGMODE *          ACBR RTCIC3
RANGE RTCIC2 FROM T001 TO 20
RANGE RTCIC3 FROM T001 TO 50
```

A session starts with application TESTCICS, using logmode D4B32782, and is allocated ACB T001 from the RANGE RTCIC2. Then, if a session starts with logmode D4B32785, RTCIC3 is used – ACB T001 is marked as unavailable so ACB T002 is used.

If a session is terminated which used an ACB obtained from a range and restarted sometime later, the ACB allocated is not necessarily the same as the previous one; the next free ACB is used.

Method 4: Multiple APPL statements per application

A range is specified on a real APPL statement as for method 3. The name of this real APPL statement must be the real VTAM applid. Additional dummy APPL statements are defined, with different characteristics – different scripts, MISER and PCTransfer settings, for example. These dummy APPL statements must not specify a RANGE and must refer to the real APPL statement by specifying its name on the APPLID parameter. The real APPL statement must not specify an APPLID parameter.

Sessions should refer to the dummy APPL statements rather than the real APPL statement. This prevents Session Manager using the same ACB from the RANGE for two sessions which use the same real application but with a different APPL because of differing characteristics.

For example:

```
APPL PCCICS   PCTransfer YES   APPLID S09CICS
APPL GENCICS  PCTransfer NO    APPLID S09CICS
APPL MINCICS  MISER NO        APPLID S09CICS
APPL ACCOUNTS STARTSCRIPT ACCT APPLID S09CICS
```

```
APPL S09CICS TERMLOGMODE * ACBR RCICS
RANGE RCICS FROM T001 TO 100
SCRIPT ACCT
```

```
...
Input key enter text 'ACCT'
...
```

A user starting a session with PCCICS is allocated an ACB T001 from the shared RANGE RCICS. This session with S09CICS has PCTransfer YES specified. A user starting a session with GENCICS is allocated an ACB T002 from the shared RANGE RCICS. This session with S09CICS has PCTransfer NO specified. A session with MINCICS uses ACB T003 with MISER NO in effect. A session with ACCOUNTS uses ACB T004 using the STARTSCRIPT ACCT. All sessions are with application S09CICS sharing RANGE RCICS.

Method 5: SHAREAPPL ACB range for multiple applications

A unique ACB is allocated per USER or TERMINAL from a SHAREAPPL RANGE for all the user's sessions. A special APPL statement is defined and is specified on the SHAREAPPL parameter on the SYSTEM statement. The SHAREAPPL APPL statement specifies a range to be used and *must not* specify an APPLID parameter. Each application (for example, TSO) requires an application APPL statement. The application APPL statement specifies the APPLID SHAREAPPL name and must not contain a TERMLOGMODE parameter. The application APPL statement name must be the real VTAM applid.

The variable `t_shareacb` is set by Session Manager to indicate the ACB currently allocated to the user. An initialization script must be coded to force second and subsequent user sessions to use the ACB allocated to the user for the first session.

For example:

```
SYSTEM SHAREAPPL ISZSHARE

APPL S09TSO  DESC 'tso subarea 9'  Applid ISZSHARE  initscript one
APPL S10NETV DESC 'Netview subarea 10' Applid ISZSHARE  initscript one

APPL ISZSHARE term1 * acbrange RANGEX

RANGE RANGEX FROM S1ISZ005 TO 015

SCRIPT one
  If t_shareacb NE ' '
    Let s_acb = t_shareacb
  End
```

PARSESS YES needs to be specified on ACBs in this range to allow a single user to start two sessions with the same application; that is, if the user has two sessions with S09TSO and one with S10NETV, all three would be allocated the same ACB from RANGEX; for example, S1ISZ005.

This method would generally be used by applications which can process parallel sessions and where an association between the terminal or user and the acb is not essential. It is recommended for applications which issue CLSDST PASS to prevent session swapping.

Note If two applications which issue CLSDST PASS are auto-started and SHAREAPPL processing is used, a session swap may still occur.

Method 6: User-level session ACB

A unique ACB is allocated per USER or TERMINAL and the name of this ACB is assigned to (user-modifiable) variable `t_user_acb`. If this variable is set then it is used as the default ACB when this user starts a forward session. In other words, it takes precedence over the `SESACB` value, but may be overridden by any of the other ACB allocation methods – for example, INITSCRIPT, RANGE associated with the application's APPL definition, and so on.

The value assigned to variable `t_user_acb` may relate to:

- **A specific ACB**
Care should be exercised if using a specific ACB as the device name. In this case, there is the possibility of multiple Session Manager userids being allocated the same virtual terminal name, which would lead to potential conflicts in applications using the ACB.

- **A range of ACBs**

In order for the value to be related to a range of ACBs, it should refer to the name of a special Session Manager APPL definition for which there is an associated RANGE definition. (If no matching APPL definition is found then the value will be assumed to be a specific ACB name.)

If the value assigned to `t_user_acb` is the name of an APPL definition then a free ACB in the associated RANGE definition will be selected automatically and variable `t_user_acb` updated to contain the name of the selected ACB. In addition, the value of the read-only variable `t_user_appl` will be set to the name of the original APPL definition. The ACB allocated to a user will be released – that is, made eligible to be selected by another user – either when the user logs off or when the value of variable `t_user_acb` is modified.

How to set variable `t_user_acb`

Note By default, variable `t_user_acb` will be blank, in which case user-level session ACB processing will not operate and the defined method for ACB allocation will be used.

Typically, variable `t_user_acb` is set in one of these ways:

- For all users, it can be set in the E22 (signon completion) exit point or script. (For details on this exit, see ‘Signon completion exit point (E22)’ on page 221).
- For TN3270E users, if a device name is supplied by the TN3270E client when the connection is established, and the SYSTEM parameter `TN3270E_CONNECT` is set to Y, then it will be set automatically. (For details on the `TN3270E_CONNECT` parameter, see the *Technical Reference*.)

For more information on setting variable `t_user_acb` in each of these ways, see ‘Examples’ on page 173.

Examples

Here is an APPL definition with an associated RANGE definition:

```
APPL SPECIAL TERML * ACBRANGE RANGEA
RANGE RANGEA FROM R001 TO 099
```

If the E22 exit assigns a value of SPECIAL to `t_user_acb` then Session Manager automatically locates the first available ACB in RANGEA and updates `t_user_acb` to contain that ACB name. This ACB will become the default for any sessions subsequently started. However, if the E22 exit assigns a value of ACBUSERA to `t_user_acb` then, assuming that there is no APPL definition called ACBUSERA, sessions subsequently started will by default use the ACB name ACBUSERA.

Similarly, if a TN3270E user connects passing SPECIAL as the device name then Session Manager allocates the first available ACB in RANGEA to that user. When the user signs on to Session Manager, variable `t_user_acb` will contain that ACB name. This ACB will become the default for any sessions subsequently started, unless overridden by another ACB allocation method. However, if a TN3270E user connects passing ACBUSERA as the device name then, assuming that there is no APPL definition called ACBUSERA, sessions subsequently started will by default use the ACB name ACBUSERA.

Notes

- 1 If a specific ACB name is assigned to variable `t_user_acb`, whether in the E22 exit or at TN3270E connection time, *no checking is performed that the ACB is unique to that user*. Therefore, it is your responsibility to ensure that multiple users are not able to start sessions to the same application using the same ACB if that will cause a conflict. The recommended method is to specify that the ACB be allocated from a range.
- 2 As mentioned above, the 't_user_acb method' of ACB allocation is overridden if any of the other ACB allocation methods is specified at a lower level.

To force use of the virtual terminal ACB for a particular session, use an INITSCRIPT (see 'Method 9: Initscript' on page 175) to reassign the value of `s_acb` back to the value of `t_user_acb`, for example:

```
IF T_USER_ACB NE ' '
  LET S_ACB = T_USER_ACB
END
```

**Method 7:
Direct user input**

A PANEL statement for a Menu screen may have a modifiable field specified for the ACB name variable `s_acb.nnnn`, where `nnnn` is the session detail number. This method should not be combined with use of a standard range.

```
PANEL menu
...
Content
...
Let sub1 = 1
Do for smax
field s_acb.sub1(8)          in,unprot,norm
...
End
...
```

**Method 8:
Menu TPSL**

The Panel used for the Menu may contain panel processing logic to supply an ACB name. This method should not be combined with use of a standard range.

```
PANEL menu
...
Content
...
Let sub1 = 1
Do for smax
If s_applid.sub1 = 'TS0'
Let s_acb.sub1 = 'ACBTS0'
End
field s_acb.sub1(8)          prot,norm,out
...
End
...
```

**Method 9:
Initscript**

An INITSCRIPT may be set up to modify the ACB name before the session start with the application. This method should not be combined with use of a standard range.

```
APPL TSO INITScript TSO

SCRIPT VM
  Let s_acb = 'A&t_user&'
```

The above example would mean that a user BILL would use an ACB ABILL, for all sessions with TSO.

**Method 10:
User exit point
E31**

ACB selection The user exit call point E31 Slave Session Initiation may be coded to set the ACB name. The session information (address 6) contains the ACB name and this is a modifiable field. If using an E31 exit script (SCRIPT EXIT31), the variable 'ec31_acb' is supplied and may be updated with the *acbname*. This method should not be combined with use of a standard range.

UNBIND WAIT applications

Overview

Some VTAM applications, such as the IBM Managed Network Service (MNS), are organized in such a way that the application name specified in the VTAM definitions is that of the application's Logon Manager. Session Manager considers this to be the primary application. After logon, control is transferred, sometimes via a Menu Selection application, to a secondary or destination application.

The transfer is achieved by sending a message to Session Manager called UNBIND with BIND forthcoming. This means that the current application has finished with the virtual terminal, but another application will acquire it by issuing an unsolicited BIND. This is handled automatically by Session Manager.

Unfortunately, a problem can occur at logoff from the secondary application, because this application may be unaware that the session should be returned to the Logon Manager or Menu Selection application. It therefore issues an UNBIND without specifying BIND forthcoming, and so Session Manager assumes that the session has ended and removes the logical terminal. The unsolicited BIND from the Logon Manager cannot be matched with a terminal id and an error message is issued.

Waiting for a BIND

The problem described previously can be overcome by requesting Session Manager to wait for the unsolicited BIND. This is achieved by specifying UNBIND WAIT in the configuration statements. It requests that Session Manager should treat all UNBINDs from non-primary applications as UNBIND with BIND forthcoming. Session Manager then maintains the session and waits for up to one minute for the unsolicited BIND to arrive.

An UNBIND from a primary application is treated as it is and causes normal session termination.

Session termination from a secondary application

In the case of MNS, the user is able to specify, via the AUTO-LOGOFF parameter in the MNS profile, whether logging off the secondary application should return the session to the Logon Manager or the Menu Manager. If the user has been configured to return to the Menu Manager, which has a different VTAM applid to the Logon Manager which is the primary application, and then wishes to terminate the session, Session Manager VTAM processes the UNBIND as an UNBIND with BIND forthcoming, if UNBIND WAIT has been specified. The session then waits for one minute before terminating.

The function of the UNBINDAPPL parameter is to permit the specification of an application name, in addition to the primary application name, for which an UNBIND will not be processed as an UNBIND with BIND forthcoming, and the session will then terminate immediately.

UNBIND processing example

Assume the Session Manager configuration specifies:

```
APPL MYEXAMPLE
DESC 'To demonstrate UNBIND WAIT and UNBINDAPPL'
APPLID LOGMAN
UNBIND WAIT
UNBINDAPPL MENUMAN
```

The console messages generated might be:

- 1 The user starts the session with the Logon Manager:

```
ISZSAA261I GS SA5L022 SESSION LOGMAN STARTING -  
LOGMODE=D4B32795ACB=ISZ001
```

- 2 The user logs on successfully, the Logon Manager then sends an UNBIND(BIND FORTHCOMING), and Session Manager VTAM receives a BIND from the Menu Manager:

```
ISZSAA271I GS SA5L022 SA5GS001 SESSION PASSED FROM LOGMAN TO  
MENUMAN
```

- 3 After a selection by the user, the Menu Manager issues an UNBIND(BIND FORTHCOMING), which is followed by a BIND from the destination application:

```
ISZSAA271I GS SA5L022 SA5GS001 SESSION PASSED FROM MENUMAN  
TODESTAPPL
```

- 4 The user terminates the destination application, which sends an UNBIND. Since UNBIND WAIT is specified, it is treated as an UNBIND (BIND FORTHCOMING) and Session Manager waits for the BIND from the Menu Manager, which duly arrives:

```
ISZSAA271I GS SA5L022 SA5GS001 SESSION PASSED FROM DESTAPPL  
TOMENUMAN
```

- 5 The user logs off the Menu Manager. Since UNBINDAPPL MENUMAN has been specified, the UNBIND received by Session Manager is not treated as an UNBIND(BIND FORTHCOMING) and the session is terminated:

```
ISZSAA262I GS SA5L022 SESSION LOGMAN ENDED
```

Additionally, message 700 is output when a Bind arrives, message 701 is output when an Unbind arrives, message 702 is output when a Timeout occurs.

These messages are used for debugging but may be suppressed from the console display by specifying MESSAGE 700 LOG NO, and so on, in the configuration.

Best fit ACB allocation for RANGE statements

When various TERMLOGMODE logmodes are specified on an APPL statement, Session Manager tries to select the RANGE using the logmode generated for the session. If this logmode is not specified, or if the selected range is fully allocated, Session Manager selects the next most suitable logmode entry (using the tables for each device shown in the following section) and checks it with the TERMLOGMODE logmodes. If it does not exist, or its RANGE is full, the next logmode entry is checked. This continues until a logmode is found and an ACB is allocated from its RANGE. If Session Manager reaches the end of the table and an ACB has not been allocated, the session is not established and an error message is issued.

When a TERMLOGMODE parameter on an APPL statement contains an asterisk (“*”) instead of a logmode entry, the logmode generated for the session is used. This is a means of selecting a RANGE to be used for all terminal types where a specific entry match has not been found. Refer to the tables for when this is used.

The success of the ‘next best fit’ technique is largely dependent upon the Installation defining ranges of ACBs for the suitable logmode entries, as well as the logmode entries defined for the terminals in the Installation. Refer to the example following the tables of next best fit logmode entries.

The `t_logm_bf` variable contains the name of the best fit logmode for the terminal. The `t_logm` variable contains the logmode that was used when the user signed on.

Tables of next best fit logmode entry names

This section contains two tables which show, for each terminal type, the logmode entry which is selected by default, and the names that are subsequently used when no ACB for that logmode entry is available. The first table is for non-SNA terminals. The second is for SNA terminals.

- The Term type column is the initial terminal type (logmode) which is being used to establish the session (for example, 3278-2) which is a model 2 non-SNA, non extended feature terminal.
- The Next best term type column is a list which Session Manager searches to find one most like the original terminal type.
- The Logmode entry column shows the standard IBM logmode entry name which Session Manager uses to establish the session for that type of terminal. If the Installation needs to override these standard names, refer to the SYSTEM statement LOGMnnx parameters in the *Technical Reference* manual.

Table 1: Non-SNA

Term type	Next best term type	Logmode entry	Term type	Next best term type	Logmode entry
3277-1		S3270	3277-2		S3270
	3278-1	D4B32781		3278-2	D4B32782
	3278-1 SNA	D4A32781		3278-2 SNA	D4A32782
	Default *			Default *	
	3278-2	D4B32782			
	3278-2 SNA	D4A32782			

Term type	Next best term type	Logmode entry	Term type	Next best term type	Logmode entry
3278-1		D4B32781	3278-2		D4B32782
	3278-1 SNA	D4A32781		3278-2 SNA	D4A32782
	3277-1	S3270		3277-2	S3270
	Default *			Default*	
3278-3		D4B32783	3278-4		D4B32784
	3278-3 SNA	D4A32783		3278-4 SNA	D4A32784
	Default *			Default *	
	3278-2	D4B32782		3278-2	D4B32782
	3278-2 SNA	D4A32782		3278-2 SNA	D4A32782
	3277-2	S3270		3277-2	S3270
3278-5		D4B32785	3278-X /3193		D4B3278X
	3278-5 SNA	D4A32785		3278-X SNA	D4A3278X
	Default *			Default *	
	3278-2	D4B32782		3278-2	D4B32782
	3278-2 SNA	D4A32782		3278-2 SNA	D4A32782
	3277-2	S3270		3277-2	S3270
3279-2		D4B32792	3279-3		D4B32793
	3279-2 SNA	D4A32792		3279-3 SNA	D4A32793
	Default *			Default *	
	3278-2	D4B32782		3278-3	D4B32783
	3278-2 SNA	D4A32782		3278-3 SNA	D4A32783
	3277-2	S3270		3279-2	D4B32792
				3279-2 SNA	D4A32792
				3278-2	D4B32782
				3278-2 SNA	D4A32782
				3277-2	S3270
3279-4		D4B32794	3279-5		D4B32795
	3279-4 SNA	D4A32794		3279-5 SNA	D4A32795
	Default *			Default *	
	3278-4	D4B32784		3278-5	D4B32785
	3278-4 SNA	D4A32784		3278-5 SNA	D4A32785
	3279-2	D4B32792		3279-2	D4B32792

Term type	Next best term type	Logmode entry	Term type	Next best term type	Logmode entry
	3279-2 SNA	D4A32792		3279-2 SNA	D4A32792
	3278-2	D4B32782		3278-2	D4B32782
	3278-2 SNA	D4A32782		3278-2 SNA	D4A32782
	3277-2	S3270		3277-2	S3270
3279-X /3193		D4B3279X	3279-D	Queryable Logmode	ISZDYNAM
	3279-X SNA	D4A3279X		3279-D SNA	ISZDYNAS
	Default *			Default *	
	3278-X	D4B3278X			
	3278-X SNA	D4A3278X			
	3279-2	D4B32792			
	3279-2 SNA	D4A32792			
	3278-2	D4B32782			
	3278-2 SNA	D4A32782			
	3277-2	S3270			

Table 2: SNA

Term type	Next best term type	Logmode entry	Term type	Next best term type	Logmode entry
3278-1 SNA		D4A32781	3278-2 SNA		D4A32782
	3278-1	D4B32781		3278-2	D4B32782
	3277-1	S3270		3277-2	S3270
	Default *			Default *	
3278-3 SNA		D4A32783	3278-4 SNA		D4A32784
	3278-3	D4B32783		3278-4	D4B32784
	Default *			Default *	
	3278-2 SNA	D4A32782		3278-2 SNA	D4A32782
	3278-2	D4B32782		3278-2	D4B32782
	3277-2	S3270		3277-2	S3270
3278-5 SNA		D4A32785	3278-X/3193 SNA		D4A3278X
	3278-5	D4B32785		3278-X	D4B3278X
	Default *			Default *	
	3278-2 SNA	D4A32782		3278-2 SNA	D4A32782

Term type	Next best term type	Logmode entry	Term type	Next best term type	Logmode entry
	3278-2	D4B32782		3278-2	D4B32782
	3277-2	S3270		3277-2	S3270
3279-2 SNA		D4A32792	3279-3 SNA		D4A32793
	3279-2	D4B32792		3279-3	D4B32793
	Default *			Default *	
	3278-2 SNA	D4A32782		3278-3 SNA	D4A32783
	3278-2	D4B32782		3278-3	D4B32783
	3277-2	S3270		3279-2 SNA	D4A32792
				3279-2	D4B32792
				3278-2 SNA	D4A32782
				3278-2	D4B32782
				3277-2	S3270
3279-4 SNA		D4A32794	3279-5 SNA		D4A32795
	3279-4	D4B32794		3279-5	D4B32795
	Default *			Default *	
	3278-4 SNA	D4A32784		3278-5 SNA	D4A32785
	3278-4	D4B32784		3278-5	D4B32785
	3279-2 SNA	D4A32792		3279-2 SNA	D4A32792
	3279-2	D4B32792		3279-2	D4B32792
	3278-2 SNA	D4A32782		3278-2 SNA	D4A32782
	3278-2	D4B32782		3278-2	D4B32782
	3277-2	S3270		3277-2	S3270
3279-X/ 3193 SNA		D4A3279X	3279-D SNA	Queryable Logmode	ISZDYNAS
	3279-X	D4B3279X		3279-D	ISZDYNAM
	Default *			Default *	
	3278-X SNA	D4A3278X			
	3278-X	D4B3278X			
	3279-2 SNA	D4A32792			
	3279-2	D4B32792			
	3278-2 SNA	D4A32782			
	3278-2	D4B32782			
	3277-2	S3270			

Example of next best fit ACB allocation

This section shows an sample APPL statement with four different TERMLOGMODE names available, each with a different rangename specified. Each range is also shown.

```
APPL TESTCICS TERMLOGMODE D4B32783 ACBR RTCIC3 >>> Note 1
                TERMLOGMODE D4B32784 ACBR RTCIC4 >>> Note 2
                TERMLOGMODE D4B32785 ACBR RTCIC5 >>> Note 3
                TERMLOGMODE *          ACBR RTCIC2 >>> Note 4
```

```
RANGE RTCIC2 FROM S05T002 TO 10
RANGE RTCIC3 FROM S05T011 TO 20
RANGE RTCIC4 FROM S05T021 TO 30
RANGE RTCIC5 FROM S05T031 TO 40
```

Notes

- 1 D4B32783 is a non-SNA model 3 non extended feature logmode
- 2 D4B32784 is a non-SNA model 4 non extended feature logmode
- 3 D4B32785 is a non-SNA model 5 non extended feature logmode
- 4 The '*' logmode is the 'catch-all' for any other terminals attempting to establish a session with this application, for which the best fit process does not yield a logmode entry which appears on the APPL statement.

If a 3278 model 2 non-SNA terminal tries to establish a session with application TESTCICS, the default logmode Session Manager attempts to use is D4B32782. The APPL statement does not contain a TERMLOGMODE for that logmode entry, so the best fit table entry for non-SNA 3278-2 is consulted:

```
3278-2          D4B32782
                3278-2 SNA  D4A32782
                3277-2     S3270
                Default *
```

The next best logmode is D4A32782 – no TERMLOGMODE exists so that cannot be used. The same is true for S3270. The next best logmode is the default * TERMLOGMODE type which is specified. So Session Manager uses RANGE RTCIC2 to allocate an ACB. The logmode used for the session is D4B32782.

The following table shows the results of logon requests to this application:

Terminal logged on to Session Manager	Implied logmode name	Session established	ACB range used	Best fit used?
3277-2 Non-SNA	S3270	Non-SNA 3277-2	RTCIC2	Yes
3278-2 Non-SNA	D4B32782	Non-SNA 3278-2	RTCIC2	Yes
3278-3 Non-SNA	D4B32783	Non-SNA 3278-3	RTCIC3	No
3278-4 Non-SNA	D4B32784	Non-SNA 3278-4	RTCIC4	No
3278-5 Non-SNA	D4B32785	Non-SNA 3278-5	RTCIC5	No
3278-2 SNA	D4A32782	SNA 3278-2	RTCIC2	Yes
3278-3 SNA	D4B32783	Non-SNA 3278-3	RTCIC3	Yes
3278-4 SNA	D4B32784	Non-SNA 3278-4	RTCIC4	Yes
3278-5 SNA	D4B32785	Non-SNA 3278-5	RTCIC5	Yes

Choosing a LOGMODE

Logmode entry selection and REBIND implications

When a user logs on to Session Manager, a 'Bind Image' is returned which Session Manager uses to determine the protocol by which it communicates with the physical terminal. Session Manager uses this same bind image to generate a logmode entry which defines the default session parameters for communication between Session Manager and any applications accessed from that terminal.

A complete list of the logmode entry names which Session Manager generates can be found in the description of the LOGMnnX parameters of the SYSTEM statement. The Session Manager names are taken from the IBM supplied logmode entry names. The names used may be altered on those parameters if the Installation logmode names differ from the Session Manager names.

Note Remote SNA terminals will use LOCAL SNA logmodes for application sessions.

The fields which Session Manager examines in the bind image and that enable it to determine which logmode entry to use are:

- The LU type. (Session Manager only supports LU0 (non SNA) and LU2 (SNA) type devices).
- The Query flag in the presentation services portion (PSERVIC).
- The default and alternate screen sizes in the PSERVIC.

Session Manager provides the option of using *either* the logmode supplied *or* a logmode which is appropriate to the terminal type. REBIND YES, which is the default, causes Session Manager to bind with a logmode that appears to be the most suitable. To use the supplied logmode, REBIND NO must be specified in the configuration file.

When REBIND YES is in effect, Session Manager attempts to use the physical characteristics of the terminal, ignoring any logmode supplied at logon. This means that if, for example, a controller was recabled, resulting in a terminal being plugged into a port formerly used by a printer (and thereby having a logmode for a printer), the session with the terminal would be rebound with a suitable logmode for the terminal device.

The 3270 extended feature code bit is always set on in the bind image, so any extended features are automatically used if they exist. The session parameters in the bind image are altered so that the alternate screen size is obtained from a WSF Read Partition (Query) response. This size is then used in the bind when an OPNDST request is issued. If successful, the logon is complete. If the session parameters are invalid, then different screen sizes are tried in an attempt to find a suitable size. Once the bind is successful, the logon is complete.

When REBIND NO is in effect, Session Manager makes no attempt to bind with a logmode other than the one supplied. Terminals therefore have all the characteristics specified by the supplied logmode. However, the name used is the standard name as used by Session Manager. If the session parameters are invalid, an error message is issued and the logon is rejected. For example, if a non-SNA terminal logs on with an SNA logmode (for example, D4A32794) the logon will be rejected with an 0821 sense code.

If REBIND NO is specified for a terminal and the logmode supplied specifies WSF Read Partition (Query), or the extended data stream bit is set on, Session Manager uses REBIND YES for Session Manager panels and uses the supplied logmode for application sessions.

Application session logmode entry selection

When a user starts a session with an application, the methods to determine which logmode is used are:

- 1 The application session parameters are derived from the Bind between Session Manager and the physical terminal after REBIND processing. This is the default action if no logmode is supplied.
- 2 A LOGMODE parameter may be specified for the session on the TERMINAL, USER, PROFILE or APPL statement. This overrides the default name.
- 3 A logmode name can be entered at the Menu screen prior to selecting the session. To use this method, the panel definition must have an input field for the variable s_logm.nnnn, where nnnn is the session detail number. This overrides the default and the LOGMODE parameter setting.
- 4 A logmode name can be set by an INITSCRIPT using Session Manager Panel and Script Language. This overrides any previous setting from the menu or the configuration.
- 5 The user exit entry point E31, slave session initiation, may set the s_logm.nnn variable. This overrides all the methods outlined above.

If a RANGE is specified for an application, Session Manager may need to derive a best fit logmode name based on the logmode established above.

Note The session parameters used to establish a session do not have to reflect the physical terminal characteristics. As an illustration of this, suppose an Installation has an application which cannot handle SNA terminals of any kind. Normally, no SNA terminal could use such an application, but with Session Manager, this is no longer the case.

Using method 2 as described above, for example:

```
PROFILE SPECIAL
  SESSION 1
  APPLID    BSCAPPL
  DESC      'A NON SNA APPLICATION'
  LOGMODE   D4B32782
```

This causes Session Manager to always use logmode D4B32782 (which is non SNA) to establish a session with the application 'BSCAPPL' when session 1 is selected from any terminal using PROFILE SPECIAL.

MTS parameters

If Session Manager receives MTS parameters when it connects to a terminal, the parameters are passed to any application started at the terminal. These parameters were first available with VTAM 3.3.

The parameters enable for each terminal: the associated main printer; the associated secondary printer; and the VDU model to be defined in one place; that is, with the VTAM configuration statements defining the terminal. Providing the application extracts these fields, it does not need to have its own associated printer definitions.

The definitions are supplied to a VTAM application in the 'LU DEFINITION CONTROL VECTOR' in the CINIT. If the associated printers are not in the same domain as Session Manager, the definitions are not passed to the applications.

If an Installation wishes to prevent or alter the MTS parameters being passed to sessions from the terminal MTS data, the variables `t_mts_prt1`, `t_mts_prt2` and `t_mts_mod1` are read-only and contain the terminal values. The variables `s_mts_prt1`, `s_mts_prt2` and `s_mts_mod1` are updateable. They contain the terminal values by default. MTS data can be specified for the session even when no terminal MTS data was supplied by VTAM.

User-level session ACB support

This section describes the support provided by Session Manager that enables an Installation to use one virtual terminal name for all of a user's forward sessions – for example to CICS, TSO, and so on.

Overview of facilities

User-level session ACB support in Session Manager enables an Installation to use one virtual terminal name for all of a user's forward sessions.

Note The ACB name takes precedence over the `SESACB` value, but it may be overridden at session level by any of the other Session Manager ACB selection mechanisms – for example, `INITSCRIPT`, `RANGE` associated with the application's APPL definition, and so on.

The support is implemented using these user-level variables:

Variable	Description
<code>t_user_acb</code>	A user-modifiable variable, which contains the name of the current user's virtual terminal ACB, if any.
<code>t_user_appl</code>	A read-only variable, which contains the name of the APPL statement used to nominate the ACB range from which the name of the virtual terminal ACB was allocated.

How to set variable `t_user_acb`

Note By default, variable `t_user_acb` will be blank, in which case user-level session ACB processing will not operate and the defined method for ACB allocation will be used.

Typically, variable `t_user_acb` is set in one of these ways:

- For all users, it can be set in the E22 (signon completion) exit point or script. (For details on this exit, see 'Signon completion exit point (E22)' on page 221.)
- For TN3270E users, it will be set automatically when a TN3270E connection is established and the `SYSTEM` parameter `TN3270E_CONNECT` is set to `Y`.

In this case, the virtual terminal name will be determined from the device name optionally passed by the TN3270E client. For more information, see 'Additional support for TN3270E' on page 355.

The value assigned to variable `t_user_acb` may relate to:

- **A specific ACB**
Care should be exercised if using a specific ACB as the device name. In this case, there is the possibility of multiple Session Manager userids being allocated the same virtual terminal name, which would lead to potential conflicts in applications using the ACB.
- **A range of ACBs**
In order for the value to be related to a range of ACBs, it should refer to the name of a special Session Manager APPL definition for which there is an associated `RANGE` definition. (If no matching APPL definition is found then the value will be assumed to be a specific ACB name.)

If you set variable `t_user_acb` to the name of an APPL definition then Session Manager will attempt to locate an available ACB within the associated range. If one is found then Session Manager will update variable `t_user_acb` automatically, setting it to the name of the selected ACB, and will also set the read-only variable `t_user_appl` to the name of the original APPL definition. If an ACB in the specified range is not available then the values of variables `t_user_acb` and `t_user_appl` will be cleared.

Notes

- 1 If the value of variable `t_user_acb` was selected using an APPL statement `RANGE` then the ACB will be deallocated from the range when the user logs off and once again become eligible to be selected by another user.
- 2 Modifying the value of variable `t_user_acb` when it contains an ACB name selected using an APPL statement `RANGE` will also cause that ACB to be deallocated from the range. If the user has existing sessions that use this previous ACB, another user may subsequently select it and attempt to start sessions with the same applications using the same ACB, possibly resulting in conflicts.

For the above reasons it is recommended that, if required, variable `t_user_acb` is set only as described above, at signon time (using the E22 exit) or at TN3270E connect time.

Summary

User-level session ACB support, which enables an Installation to use one virtual terminal name for all of a user's forward sessions, is implemented using these user-level variables:

Variable	Description
<code>t_user_acb</code>	A user-modifiable variable, which contains the name of the current user's virtual terminal ACB, if any.
<code>t_user_appl</code>	A read-only variable, which contains the name of the APPL statement used to nominate the ACB range from which the name of the virtual terminal ACB was allocated.

Variable `t_user_acb` is typically set in one of these ways:

- In the E22 (signon completion) exit point or script.
- Automatically, when a TN3270E connection is established and the SYSTEM parameter `TN3270E_CONNECT` is set to Y.

The Virtual Terminal Masking facility

The Session Manager Virtual Terminal Masking (VTM) facility allows the ACB name for a selected session to be assigned according to the APPL name/userid combination. It provides a panel-driven means of 'masking' a virtual terminal ACB to a site's requirements without the need to write your own code and execute it through an INITSCRIPT.

Overview

The Virtual Terminal Masking facility provides a means of building the ACB name for a selected session using any combination of characters from the userid and session user's terminal name, dependant on the APPL /user combination. This is accomplished by Session Manager maintaining a 'mask' that is held against the APPL name/userid combination. The APPL name/userid/mask relationship is held on a PDSE with a DD name of VTM.

VTM entries are maintained via a set of scripts and panels.

Note Escapes are not permitted from some panels (in which the command line prefix is shown as --->). This is to allow for an escape sequence to be entered as *data*. Those panels where escapes *are* allowed have the usual format (====>) for the prefix.

Specifying VTM scripts

You need to specify an INITSCRIPT of ISZSVTM for sessions to which you wish virtual terminal masking to be applied. ISZSVTM is a supplied script which will resolve the ACB name by interrogating the VTM PDSE entries at session selection. If no entries exist for the APPL name/userid combination then the ACB name will be resolved from the configuration.

The INITSCRIPT can be specified on the SYSTEM statement so that it applies to all sessions which do not have an INITSCRIPT at the USER/TERMINAL/PROFILE level. If additional INITSCRIPT processing is required then ISZSVTM can be called by another INITSCRIPT.

Maintaining VTM entries

You access the VTM facility by selecting a session from the main menu. The facility is used to define and maintain VTM entries and, by default, access is only allowed to users with an Authority Level of 9.

Display list of APPL entries

On selecting the VTM facility you are presented with the **VTM Appl List** panel which displays an alphabetical list of APPL names.

The maintenance tasks include:

- List existing APPL entries.
- Copy an existing APPL entry.
- Modify an existing APPL entry.
- Rename an existing APPL entry.

- Delete an existing APPL entry.

VTM Appl List panel

The input fields are blank when the **VTM Appl List** panel is first displayed. You can tailor and control the entries displayed using **FILTER** and **FIND** commands.

Each entry consists of an input action field, the entry name (APPL name) and an input field to contain the name of a new entry when using the **COPY** or **RENAME** line command.

The panel behaves in a similar manner to the **OLA APPL list** (see *Online and Batch Administration* manual for details).

APPL names can be generic (for example **CICS***, with that entry applying to all APPL names prefixed with **CICS** and which do not match a more specific entry).

Filtering entries

To subset the entries on the **VTM Appl List** panel, type your selection pattern in the **Filter** input field, or leave blank to select all APPLs. Wildcard entries may be used:

- * An asterisk indicates any number of any characters.
- + A plus sign indicates any single character in that position.

Example Selects

- *ABC Names ending in ABC.
- *A* Names containing a A, possibly surrounded by any other characters.
- ABC* Names beginning with ABC.
- ++ABC Five-character names ending in ABC.
- +A* Names at least 2 characters in length with A as the second character.
- ++A* Names of at least 3 characters in length with A as the third character.

Multi-page configuration definition lists

For a multi-page APPL list, scroll indicators are displayed in the bottom right hand corner of each page:

- >>>> Indicates you can page forward to the next page of entries.
- <<<< Indicates you can page backward to the previous page of entries.

Codes for List actions

Action codes may be entered as line commands for required APPL entries:

- C
Copy an APPL entry.
- D
Delete an APPL entry.
- E
Select/edit an APPL entry.

R

Rename an APPL entry.

S

Select/edit an APPL entry.

Modifying a particular APPL entry

To modify a particular APPL entry select it and press Enter or use the E (= Edit) or S (= Select) action code (see above).

Processing of attributes for a particular APPL entry is similar to processing of common end user attributes (see *Online and Batch Administration manual*).

Display list of users

On selecting an APPL on the **VTM Appl List** panel you are presented with the **VTM User List** panel which displays an alphabetical list of user entries.

The maintenance tasks include:

- List existing user entries.
- Copy an existing user entry.
- Modify an existing user entry.
- Rename an existing user entry.
- Reset a user entry to its original value or restore a deleted entry.
- Delete an existing user entry.

VTM User List panel

You can tailor and control the entries displayed using FILTER and FIND commands.

Each entry consists of an input action field, the entry name (userid) and an input field to contain the name of a new entry when using the COPY or RENAME line command. Each user entry will display, alongside it, an ACB mask.

As with APPL names, user names can be generic (for example, ABC*, which would apply to all user entries prefixed ABC).

Filtering entries

This works in a similar way to the **VTM Appl List** panel (see 'Filtering entries' on page 190).

Multi-page configuration definition lists

This works in a similar way to the **VTM Appl List** panel (see 'Multi-page configuration definition lists' on page 190).

Codes for List actions

Action codes may be entered as line commands for required APPL entries:

C

Copy a user entry.

D	Delete a user entry.
E	Select/edit a user entry.
R	Rename a user entry.
S	Select/edit a user entry.
Z	Reset a user entry to its original value or to restore an entry deleted during the update session.

Saving changes to user entries

Press PF3 (Save) to save any amendments made to the user entries of an APPL and return to the **VTM Appl List** panel. Or press PF12 (Cancel) to return to the **VTM Appl List** panel without saving any amendments made.

Maintain user mask

The mask value for an entry can be modified by selecting the user entry on the **VTM User List** panel. This displays the **VTM Update User Mask** panel which allows you to edit the mask to be used in the building of the ACB name.

VTM Update User Mask panel

The **VTM Update User Mask** panel displays an **ACB Mask** field for a given user. The mask can contain either a VTAM ACB name or a Virtual Terminal mask.

A mask contains one or more masking characters. Valid masking characters and their use in ACB name substitution are shown below:

Mask Value	ACB Name Substitution
Any alphanumeric character	To be used implicitly.
The character ‘%’	Use the corresponding character from the userid
The character ‘?’	Use the corresponding character from the VTAM user’s terminal name (luname).
‘&tps1variablename&’	Use the variable value.
‘&tps1variablename(2,3)&’	Use a sub-set of the TPSL variable value from position two for a length of three.

So given the following:

userid = US123, and

luname = SVTRM999

then a mask of ‘IS%%%???’ or ‘IS&t_user(3,3)&???’ would generate an ACB name of IS123999.

Saving changes to user mask

Press PF3 (End) on the **VTM Update User Mask** panel to return to the **VTM User List** panel and then, to save your changes, press PF3 on that panel; or press PF12 to cancel.

Implementing the VTM facility

A sample APPL ISZAVTM is supplied in the Classic member ISZCAPPL in library .SISZCONF, or in library .SISZSAPL for OLA. Refer to 'APPL definitions' on page 163 on how to configure the VTM APPL in a Classic or OLA system. Adding a session referring to this APPL will cause that session to launch an internal session which will display the Virtual Terminal Masking facility.

VTM with multiple Session Manager instances

To control contention between VTM users attempting to modify the same item(s), it is recommended that VTM sessions operate within a single Session Manager instance. This will be the usual mode of operation, but this section describes the steps required to maintain the integrity of the VTM PDSE when shared across multiple Session Manager instances in both a Sysplex and a non-Sysplex environment.

Configuration definitions

The `vtmsinit` script will automatically determine whether the environment is Sysplex or non-Sysplex and will assume that the following changes have been implemented for the appropriate environment.

Non-Sysplex environment

To enable VTM use across multiple Session Manager instances, make these changes to the (shared) configuration:

- 1 Add the VTAM ACB name (APPLID xxxxxxxx) for the Session Manager VTM instance and the DATA parameter to the APPL definition as follows:

```
APPL iszavtm
INITSCRIPT vtmsinit
STARTSCRIPT vtmstart
APPLID xxxxxxxx
INTERNALSESS yes
DESC "VTM Maintenance"
DATA "&t_user&/&t_pass&//vtmprof"
```

- 2 Add this PROFILE definition:

```
PROFILE iszprov
autoselect 8987
SESSION 8987
quitactive yes
applid ISZAVTM
```

Note This profile is supplied as a sample in Classic member ISZCPROF in library .SISZCONF, or in library .SISZSPRF for OLA.

Ensure your users do not use session 8987; and they must not have AUTOSELECT explicitly coded.

Profile `iszprov`, which is loaded by all Session Manager instances but acted upon only by the Session Manager VTM instance, ensures that the VTM session is AUTOSELECTed for all users signing on to VTM from other instances.

The user's credentials will be passed to the Session Manager VTM instance and, if accepted, the Session Manager signon panel will be bypassed.

3 Add LINK definitions for all systems

LINK definitions

An example will be used to describe how to define LINK definitions to connect Session Manager instances to each other.

Notes

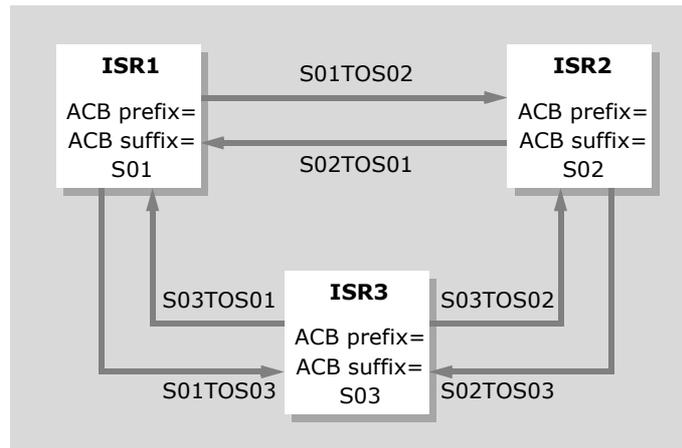
- 1 The LINK definitions are shared/loaded by all Session Manager instances that share the configuration, so the ACBs defined in each LINK definition must use `%%gc_acb_prefix%%` and `%%gc_acb_suffix%%`.
- 2 Create two LINK definitions - one for the VTM Session Manager instance and one for the local Session Manager instance. For each additional Session Manager instance, add a new LINK definition of the type described in the example below.

Example

Suppose that there are three Session Manager instances, whose LOCALNODE names are ISR1, ISR2 and ISR3. To connect these Session Manager instances to each other, you would need to perform these steps:

- 1 For each instance, define the ACBs to be used for the local ends of the links:

LOCALNODE	gc_acb_prefix	gc_acb_suffix	Local ACBs
ISR1	S01	S01	S01T0S02 S01T0S03
ISR2	S02	S02	S02T0S01 S02T0S03
ISR3	S03	S03	S03T0S01 S03T0S02



- 2 Define a single set of LINK definitions to be used by all of the Session Manager instances, using the LOCALNODE names of the instances for the LINK names:

```

LINK ISR1 VTAM %%gc_acb_prefix%%TOS01 TO
                S01T0%%gc_acb_suffix%% LOGMODE logmode
LINK ISR2 VTAM %%gc_acb_prefix%%TOS02 TO
                S02T0%%gc_acb_suffix%% LOGMODE logmode
LINK ISR3 VTAM %%gc_acb_prefix%%TOS03 TO
                S03T0%%gc_acb_suffix%% LOGMODE logmode
  
```

Note Prefixing and suffixing `gc_acb_prefix` and `gc_acb_suffix` with two consecutive percent signs (`%%`) causes their values to be substituted *before* the contents of the definition are passed to the Session Manager parser.

Consider the instance having a LOCALNODE of ISR1. For ISR1, `gc_acb_prefix` and `gc_acb_suffix` have the value S01, so these LINK definitions will be used:

```

LINK ISR1 VTAM S01TOS01 TO S01TOS01 LOGMODE logmode
LINK ISR2 VTAM S01TOS02 TO S02TOS01 LOGMODE logmode
LINK ISR3 VTAM S01TOS03 TO S03TOS01 LOGMODE logmode
  
```

Note A LINK having the same name as the LOCALNODE is *not* activated so, in this case, the LINK named ISR1 would *not* be activated.

Sysplex environment

Session Manager will automatically allocate a master Session Manager node within the Sysplex environment where all VTM updates will take place. Any LINK statements will be ignored.

In a Sysplex environment where the multiple Session Manager instances are assigned to the same SYSPLEXGroup it is mandatory that all the Session Manager instances must share the same configuration datasets.

To enable VTM across multiple Session Manager instances, make these changes to the (shared) configuration:

- 1 Add this APPL definition:

```

APPL iszavtm
DESC 'VTM Maintenance'
INITSCRIPT vtmsinit
  
```

STARTSCRIPT vtmstart

INTERNALSESS YES

INQUIRE no

Note This APPL is supplied as a sample in Classic member ISZCAPPL in library .SISZCONF, or in library .SISZSAPL for OLA.

CHAPTER 8

Session Manager user exit processing

There are several sample user exits supplied with the IBM Session Manager for z/OS system. A list of these is given in the README file (ISZ£READ) supplied on the distribution media. Alternatively, an Installation may create its own exit. This chapter describes what is required for an Installation-specified user exit.

The subjects covered in this chapter are:

- ‘Overview’ on page 199
- ‘Installing the exits’ on page 204
- ‘Standard exit conventions’ on page 205
- ‘Initialization exit point (E01)’ on page 208
- ‘Configuration statement processing exit point (E05)’ on page 209
- ‘Application status change exit point (E06)’ on page 211
- ‘Timer interval exit point (E08)’ on page 212
- ‘CALLEXIT invocation point (E09)’ on page 213
- ‘Logon exit point (E11)’ on page 215
- ‘Signon validation exit point (E21)’ on page 217
- ‘Signon completion exit point (E22)’ on page 221
- ‘Input 3270 datastream exit point (E25)’ on page 223
- ‘IDLEDISC/IDLELOGOFF/IDLELOCK timer expiry exit point (E26)’ on page 224
- ‘Signoff exit point (E29)’ on page 225
- ‘Slave session pre-initiation exit point (E31)’ on page 227
- ‘Slave session post-initiation exit point (E33)’ on page 229
- ‘Output 3270 datastream exit point (E35)’ on page 231
- ‘SIDLTIME timer expiry exit point (E36)’ on page 232
- ‘Slave session termination and SMF record exit point (E39)’ on page 233

- ‘User exit replacement exit point (E71)’ on page 241
- ‘Session switch exit point (E79)’ on page 242
- ‘Closedown exit point (E99)’ on page 245
- ‘Variable access’ on page 246
- ‘Producing statistics for NETSPY and ETE’ on page 251
- ‘Input and output 3270 datastream exit points’ on page 252
- ‘The Multiple Exit Driver’ on page 279
- ‘Exit scripts’ on page 282
- ‘User audit facility’ on page 284
- ‘Timeout exits facility’ on page 288

Overview

Session Manager provides a number of user exit points where site-specific processing can be performed. The exits can be used to perform functions such as modifying the configuration statements, preventing a specific terminal from accessing Session Manager, checking and modifying signon parameters, and preventing signon. In addition, they may be used for timestamping signon and signoff, Session Manager startup and termination, and slave session initiation and termination, which can then be used for accounting purposes. The slave termination exit may also be tailored to produce SMF records containing data to meet your installation requirements. Each exit point has an associated exit number for easy identification.

All user exits (except E26 and E36) can be implemented using Assembler or COBOL modules. Two 3270 datastream exit points have special considerations and are therefore discussed separately (see 'Input and output 3270 datastream exit points' on page 252). The remaining exits can be written in either of two ways:

- as a single load module containing multiple sections (one for each desired exit point). This is the standard option, but can sometimes result in a large module that becomes difficult to administer.
- as multiple load modules (one for each Exit point). This option requires the use of fixed module names, with the individual modules being accessed and administered by the supplied Multiple exit driver (see 'The Multiple Exit Driver' on page 279).

Most (but not all) user exit points can also be implemented using the script language (TPSL) in Session Manager. These are termed 'Exit scripts'. In cases where a user exit point has both an Exit script and a module, the script receives control first, and it can control (through return codes) whether the module should also be invoked. See 'Exit scripts' on page 282 for more information.

User Exit information is defined to Session Manager using the OPTION and Enn configuration statements (see *Technical Reference* manual). These statements specify:

- whether the site is using the single module option or the Multiple exit driver option.
- whether any exit scripts are to be used.

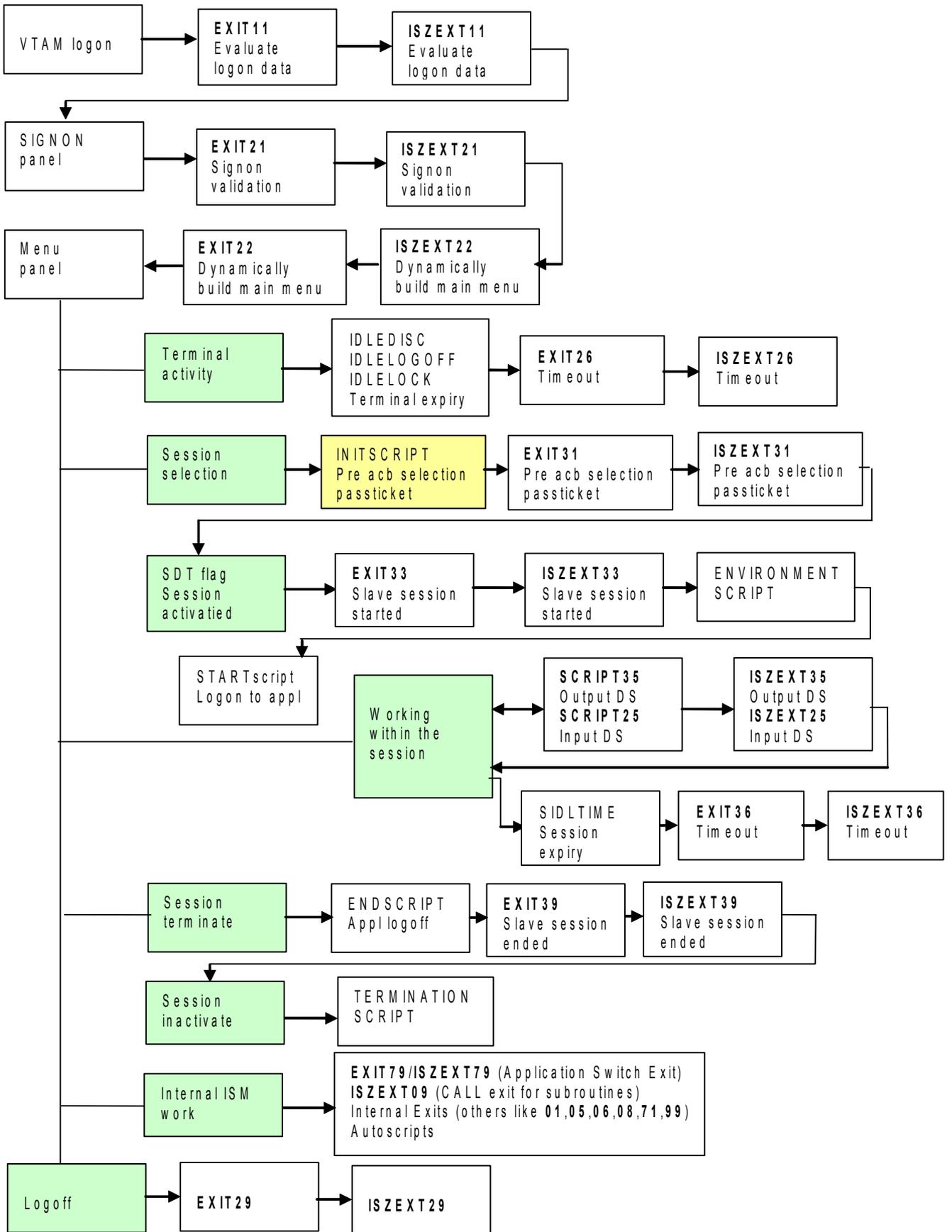
User exit points

The Session Manager user exit points are:

- Session Manager initialization time (E01)
- Configuration statement processing time (E05)
- Application status change (E06)
- Timer interval exit (E08)
- CALLEXIT invocation from a script (E09)
- At logon to the Session Manager system (E11)
- At signon validation (E21)
- At signon completion (E22)
- Input 3270 datastream exit (E25)
- On expiry of the IDLEDISC, IDLELOGOFF or IDLELOCK timers (E26)
- At logoff or disconnect from Session Manager (E29)
- Slave session pre-initiation (E31)
- Slave session post-initiation (E33)
- Output 3270 datastream exit (E35)
- On expiry of the SIDLTIME timer (E36)
- Slave session termination (E39)
- When the UPDATE command is issued, prior to replacement with a new version of the exit (E71)
- Session switch exit (E79)
- Session Manager termination (E99)

User exits flowchart

The flowchart on the following page provides an overview of Session Manager user exits functionality.



Notes

- 1 Exit points E01, E09 and E71 are only applicable to the Assembler/COBOL user exit: they cannot be specified as scripts.
- 2 Exit points E25 and E35 have special requirements – for details, see ‘Input and output 3270 datastream exit points’ on page 252.
- 3 Exit points E26 and E36 can only be specified as exit scripts.

Provided that a valid user exit name is specified as an OPTION statement parameter, the exit is automatically loaded by Session Manager at initialization time and remains resident thereafter for the duration of that session, unless reloaded as a result of an UPDATE command. Optionally, if no exit was specified at initialization, it may be loaded while Session Manager is active using the UPDATE command. See ‘Using the update facility’ in the *Technical Reference* manual for further details of this.

When an exit is loaded using the UPDATE command, the old exit is called at exit point E71. This enables the old exit to close files or perform any other necessary functions before the new exit is invoked. The exit is similarly called at Session Manager closedown at exit point E99.

The user exit is called with a comprehensive parameter list appropriate to the particular call. On each call, the exit may access any variable, either Session Manager or user-defined variable, relevant to the particular call.

Each call type is identified by a call indicator and possibly a reason code. The exit routine should therefore test for conditions relevant to the call.

Each user exit point may:

- 4 Inspect, add, modify, or delete user definable variables
- 5 Inspect or modify certain Session Manager variables
- 6 Prepare a single message for output by Session Manager to the log (message numbers 600 to 699 are available for the user exit)
- 7 With some call reasons, prepare a single message for output by Session Manager on the current users panel
- 8 Write direct to the console
- 9 Access files

For the exit to issue a message, the message must be defined in the configuration file with a number in the range 600 to 699. The text can contain, or consist entirely of, a variable portion denoted by @1 or %1. Any data that is found in the fields **log message text** or **screen message text** in the Output Message Information area pointed to by address 3 (see the Call Information layouts shown later) replaces the @1 or %1.

Note In this document, the 0x7C (that is, x’7C’) character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

For example, the exit might contain the statements:

```

MVC  EX30SMST(L'VTUBMSG1),VTUBMSG1  SET SCREEN MESSAGE
MVC  EX30SMSL,=A(50)                 SET SCREEN MSG LENGTH
MVC  EX30LMST(L'VTUBMSG1),VTUBMSG1  SET LOG MESSAGE
MVC  EX30LMSL,=A(50)                 SET LOG MSG LENGTH
LA   R2,603                           SET MESSAGE NUMBER.
ST   R2,EX30LMSN
ST   R2,EX30LMSN
. . .

```

```
VTUBMSG1 DC    CL50'Please contact TSG'
```

```

. . .
      DS    OD
EXPARM3 DSECT                * EXIT MODIFIABLE
EX30LMSN DC  A(0)             * LOG    MESSAGE NUMBER
EX30LMSL DC  A(0)             * LOG    MESSAGE LENGTH (MAX 132)
EX30LMST DC  CL132' '        * LOG    MESSAGE TEXT
EX30SMSN DC  A(0)             * SCREEN MESSAGE NUMBER
EX30SMSL DC  A(0)             * SCREEN MESSAGE LENGTH (MAX 132)
EX30SMST DC  CL132' '        * SCREEN MESSAGE TEXT
EX30LMSI DC  C'N'             * SEND MSG TO LOG (Y/N)
EX30SMSI DC  C'N'             * SEND MSG TO SCREEN (Y/N)
EX3ILMSI DC  C'N' READ ONLY * ALLOW MSG TO LOG (Y/N)
EX3SLMSI DC  C'N' READ ONLY * ALLOW MSG TO SCREEN (Y/N)
EXCLEN  EQU  *-EXPARM3       * LENGTH OF MESSAGE DSECT

```

and the configuration file would contain a definition:

```
MESSAGE 603 TEXT 'SIGNON FAILED - Password invalid. @1'
```

so the message appearing at a particular terminal would be:

```
SIGNON FAILED - Password invalid. Please contact TSG
```

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

Installing the exits

Exits are supplied already assembled and linked, but if they need modifying then this is accomplished by the assemble and link edit of the provided source members (see sample JCL `ISZEASM` in library `.SISZCONF`). These must be linked into an APF authorized library which needs to be specified in the startup `STEPLIB` concatenation for the Session Manager system.

The following exits are supplied already assembled and linked, although the source is also provided in case you should wish to modify them:

`ISZE00DR` linked as `ISZEXT00`

`ISZE09DR` linked as `ISZEXT09`

`ISZE21SF` linked as `ISZEXT21`

`ISZE22DM` linked as `ISZEXT22`

If you configure your system to use the Multiple Exit Driver (`ISZE00DR`) all available exits will be loaded at initialization of your system. The above list of supplied exits that are already assembled and linked should be reviewed and any exits not required should be removed or renamed.

Standard exit conventions

Return codes

There are a number of return codes that are used by most exit points as standard. If Exit scripts are used they *must* set the appropriate return codes for the Assembler/COBOL user exit:

Return code	Description
0	Signifies successful completion of the exit. This is applicable to all exit points.
4-16	These are specific for each exit point.
28	Signifies the exit has no processing for the function on this occasion but may wish to in the future. This should be used when the exit is to process a particular function only occasionally.
40	Suppresses further calls by the currently loaded exit for this exit point. When the exit is next reloaded, or if a return code 44 is returned by any other exit, this is reset and the exit point is called again. Any call types that the user exit cannot recognize should also return decimal 40. This is so that if future versions of Session Manager have new call types then the current exit does not have to be updated on each occasion a new version of Session Manager is installed.
44	Resets all previous exit return codes for all exit points. This can be used to reset a decimal 40 return at another exit point. The exit is effectively 'restarted' for all exit points.
52	(Exit scripts only.) Schedules the Assembler/COBOL exit with the same parameters after the Exit script terminates.

Exit load module

The user exit must be a load module, and its name must be specified on the EXIT parameter of the OPTION statement. Standard subroutine linkage conventions should be used.

For the exit to be loaded successfully, it must reside in the load library denoted by the STEPLIB statement in the startup job stream.

Register conventions

On entry to the exit, the register contents are set as follows:

Register	Contents
R1	Address of the address of parameter list addresses
R13	Register save area address

Register	Contents
R14	Return address
R15	Entry point address
R0,R2-R12	Unpredictable

Parameter list

The parameter list consists of up to eight addresses, each one pointing to an area of storage containing fields pertinent to the particular exit call. The first three addresses are similar for each exit and are reproduced below for conciseness.

- In address one the call indicator and reason code are different for each exit.
- The second address always addresses the Variable Access routine.
- The third address, output message information, always has the same format.

Address 1 – General call information

This information is applicable:

Position	Len	Description	Exit script variables	Note
1 - 3	3	Call indicator 'Enn'	ec_rcode	1
4 - 4	1	Spare		
5 - 12	8	Reason code ' <i>reason</i> '	ec_reason	2
13 - 20	8	Date in either <i>dd/mm/yy</i> or <i>mm/dd/yy</i> format	t_date	
21 - 24	4	Julian date in packed decimal (<i>00yydddF</i>)	en_julian	3
25 - 32	8	Time in <i>hh:mm:ss</i> format	t_time	
33 - 40	8	TOD clock value		
41 - 44	4	Address of statistics block		

Notes

- 1 The call indicator shown is the relevant exit number, that is, E01, ..., E99.
- 2 The reason code is different for each exit point. The relevant reason codes are given in the description of each exit point.
- 3 The julian date addressed by the Assembler/COBOL parameter list is packed; that contained in en_julian is unpacked.

Address 2 – Variable Access routine

Address 2 points to the Variable Access routine for each exit point. For details see 'Variable access' on page 246.

Address 3 – Output message information

This information may be modified by the exit.

Position	Len	Description
1 - 4	4	Log message number
5 - 8	4	Log message length
9 - 140	132	Log message text
141 - 144	4	Screen message number
145 - 148	4	Screen message date length
149 - 280	132	Screen message text
281 - 281	1	Send message to log, 'Y' or 'N'
282 - 282	1	Send message to screen, 'Y' or 'N'
283 - 283	1	Exit enabled to send message to log, 'Y' or 'N' (R/O)
284 - 284	1	Exit enabled to send message to screen, 'Y' or 'N' (R/O)
285 - 288	4	Spare

The description of each exit point shows whether a message can be output to the log, the screen, or both.

In the following descriptions of the exit points, the fields for each call are grouped in the order in which the addresses are provided in the address list. For address 1 the call indicator and reason code are given for each exit point but the other information provided by the address is the same for each exit point.

Initialization exit point (E01)

Session Manager calls this exit point when the user exit has been loaded.

OPTION parameters may be specified externally from the configuration file, that is, on the PARM field of the EXEC statement.

For any system, the OPTION statement may be placed in the configuration file. If OPTION parameters are supplied in both places then those specified in the configuration file are overridden.

The main purposes of this exit point are to enable the startup to be aborted if required, to open files, and possibly to record the time of startup.

Call information **General call information (pointed to by address 1)**

Startup call indicator 'E01'.

Reason code 'STARTUP'.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log only.

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Continue with the startup
- 4 = Abort the startup
- 28 = The user exit does not support this function

For any other value, 4 is assumed. Unlike other exit points, return codes 40 and 44 are irrelevant since the exit point is called only once.

Configuration statement processing exit point (E05)

Session Manager reads the configuration file at initialization time, and also when the Update facility has been invoked as a result of an UPDATE command in a Classic system or a PUPDATE (OLA ACTIVATE) in an OLA system. This exit point is called prior to Session Manager processing each logical record and enables the exit to modify any configuration record, and to insert records. Since each record is part of a control statement, care must be taken to avoid introducing errors, either syntactical or logical, as this may cause Session Manager to reject the whole statement.

The statement name is available in the fourth parameter to indicate to the exit the type of statement being presented. This is always the last one that Session Manager has recognized except when records from a copied member are read, in which case the statement keyword is blank until Session Manager encounters a statement name. It is also blank when the record after a COPY statement is passed to the exit for processing. For example, for the following configuration statements:

```
Profile P1
  menu commenu
Copy member1
System signonpanel logo
  logdisc exit
```

and assuming that 'member1' contained:

```
User u1
User u2
User u3
```

the contents of fields and the sequence in which records are passed to the exit would be:

Logical record image	Statement keyword
Profile P1	
menu commenu	Profile
Copy member1	Profile
User u1	
User u2	User
User u3	User
System signon panel logo	
Logdisc exit	System

Comment records at the beginning of a copied member are considered as part of the record yet to be built, but in other instances comment records are considered to belong to the preceding statement.

When a logical record is to be inserted, it should be noted that it is inserted before the current logical record.

Note OPTION statements are passed to the exit at this point except for the one that specifies the user exit name.

Call information **General call information (pointed to by address 1)**

Configuration statement call indicator 'E05'.

Reason code 'CONFIG' or 'UPDATE'.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log only.

Configuration statement information (pointed to by address 4)

Position	Len	Description	Exit script variables
1 - 24	24	Member name	ec05_name
25 - 40	16	Statement keyword, or blank	ec05_keywd

Logical record image (pointed to by address 5)

Position	Len	Description	Exit script variable
1 - 80	80	Logical record image (may be modified by the exit)	ec05_record

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Process the current logical record, which may have been modified by the user exit.
- 4 = Ignore the current logical record.
- 8 = Ignore the statement of which the current logical record is a part.
- 12 = As for return code 0, then call the exit again with the original, unmodified record. Enables new logical records to be inserted.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous exit return codes for all exit points.

For any other value, 4 is assumed.

Note The E05 user exit cannot set a return code of 8 or 12 when it processes a configuration record containing an INSTALLSU statement. If the exit does set a return code of 8 or 12 in this circumstance, the return code will be converted to a return code of zero.

Application status change exit point (E06)

This exit point is supplied for use with application status related functions. Each time the status changes for an application, this exit point is called, provided that the previous return code was not 40 and that the APPL definition is INQUIRE YES, either by default or specification.

Call information **General call information (pointed to by address 1)**

Application status change call indicator 'E06'.

Reason code 'APPLSTAT'.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log only.

Application status information, read only (pointed to by address 4)

Position	Len	Description	Exit script variables
1 - 8	8	Application name	ec06_appl
9 - 9	1	New status, 'Y' = appl available 'N' = appl unavailable	ec06_nstatus
10 - 10	1	Previous status, 'Y' = appl was available 'N' = appl was unavailable ' ' = status was not known	ec06_ostatus

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = The exit point has successfully completed.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous exit return codes for all exit points.

For any other value, 0 is assumed.

Timer interval exit point (E08)

This exit point is called every 10 minutes after Session Manager has completed initialization, and is intended for interval related functions.

Call information **General call information (pointed to by address 1)**

Access validation call indicator 'E08'.

Reason code 'INTERVAL'.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log only.

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = The exit point has successfully completed.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

For any other value, 0 is assumed.

CALLEXIT invocation point (E09)

This exit point is invoked when the CALLEXIT function call is issued by a script. An eight character user-defined reason code and up to 255 bytes of user data may be specified with the CALLEXIT function.

The supplied sample ISZE09DR is an E09 exit driver and enables multiple E09 exits to run concurrently. It dynamically loads and maintains a table of E09 exits. It requires that these exits are called ISZExxxx, where xxxx is the first four characters of the reason code. For example, an exit called ISZEWINS would be invoked for CALLEXIT reason codes of WINSADD, WINSUPD, WINSDEL and so on. The supplied member ISZCEXIT contains instructions on how to assemble and link all of the supplied sample exits.

Along with the reason code and user data, the value of the variable `t_result` is also available to the exit without the need to call the Variable Access routine. This allows an additional numeric value to be passed. This value can be modified by the exit and will be copied to the variable `t_result` upon completion of exit processing. This allows the exit to set a return code for the calling script.

Call information **General call information (pointed to by address 1)**

Script call indicator 'E09'.

User-defined reason code. This is specified on the RSN subparameter of the CALLEXIT parameter. It is blank if RSN is omitted.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log only.

Terminal session information (pointed to by address 4)

Position	Len	Description
1 - 4	4	Terminal task type
5 - 12	8	Terminal name
13 - 16	4	Terminal session id
17 - 24	8	Logmode name (VTAM only)
25 - 51	26	Bind image (VTAM only)

Final user information (pointed to by address 5)

Position	Len	Description
1 - 8	8	User name
9 - 16	8	Password

Position	Len	Description
17 - 24	8	Profile
25 - 26	2	Authority
27 - 32	6	Spare

User data (pointed to by address 6)

Only the first field can be modified, the rest are read-only.

Position	Len	Description
1 - 4	4	Value of the variable <code>t_result</code> in fullword binary format. Its value at completion of the exit is copied to <code>t_result</code> for use in the script.
5 - 5	1	Length of user data as specified by the DATA sub-parameter of the CALLEXIT parameter. It is 0 if DATA is omitted.
6 - 261	255	User data

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Exit processing was successful.
- 28 = The exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

For any other value, 28 is assumed.

Logon exit point (E11)

This exit point is called when a user logs on to the Session Manager application, before the Signon panel is displayed. Session Manager determines if there are any associated TERMINAL and PROFILE statements, and passes some of the details to the user exit.

The main function of this exit point is to either allow or reject access to Session Manager from a specific terminal.

Call information **General call information (pointed to by address 1)**

Access validation call indicator 'E11'.

Reason code:

'LOGON ' VTAM terminal.

'SIMLOGON' Simulated logon at a VTAM terminal.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log only.

Terminal session information (pointed to by address 4)

Position	Len	Description	Exit script variables	Note
1 - 4	4	Terminal task type	ec11_ttype	
5 - 12	8	Terminal name	ec11_term	
13 - 16	4	Terminal session id		
17 - 24	8	Logmode name (VTAM only)	ec11_logm	
25 - 51	26	Bind image (VTAM only)	ec11_bind	1
52 - 307	256	VTAM logon data (VTAM only)	ec11_logd	

Note

1 Access using TPSL MASKxx condition.

Predefined profile information (pointed to by address 5)

Position	Len	Description	Exit script variables
1 - 8	8	Profile name	ec11_prof
9 - 9	1	Signon required, 'Y' or 'N'	ec11_sign

Signon parameter information, may be modified by the exit (pointed to by address 6)

Position	Len	Description	Exit script variables
1 - 8	8	Userid	ec11_user
9 - 16	8	Password	ec11_pass
17 - 24	8	New password	ec11_npass
25 - 32	8	New profile	ec11_nprof
33 - 33	1	Signon required, 'Y' or 'N'	ec11_nsign
34 - 41	8	New signon panel	ec11_signp

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Allow this terminal to access Session Manager.
- 4 = Reject the logon.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

For any other value, 4 is assumed.

Note There are implications in using the VTAM logon data for determining the signon panel name in this exit.

In current releases of Session Manager the VTAM logon data is assumed to consist of a number of fields separated by forward slash (/) but possibly there may be only one field, in which case there would be no slash. This field would be assumed to be the userid and is forwarded to the E11 exit in the variable ec11_user. It is the responsibility of this exit to clear the field or variable if this is not what is intended. For information on the use of logon data to Session Manager please see 'Step 7: Signing on to Session Manager' on page 67.

Signon validation exit point (E21)

This exit point is invoked each time a user attempts to sign on to the system. It may be used to perform the security clearance checking for that user, and therefore may be used to prevent unauthorized personnel from obtaining access to the Session Manager system. Sample exits for using VSAM authorization or an ESM are supplied with Session Manager. See ‘Defining security and implementing dynamic menus’ on page 289.

Notes

- 1 If it is necessary for the E21 exit point to establish the status of a particular userid then the QUSER command (see the *Technical Reference*) can be issued from the exit point. The response to this command is a message which is returned in the `t_message` variable.
- 2 In a Parallel Sysplex environment, if a user is already signed on to (or disconnected from) a particular Session Manager node then if the user attempts to sign on again a sample E21 exit script provides a mechanism to transfer the signon to the original Session Manager node. For details, see ‘Sample signon validation exit script (E21)’ on page 422.
- 3 For Session Manager v3.1.00 and higher, a new sample E21 exit script is supplied. Script ISZE21PH provides all the functionality of the current E21 assembler exit and additionally supports:
 - External Security Manager password phrases.
 - Multi threaded ISM signon.
 - The ability to override any multiple profiles assigned during user configuration, up to a maximum of eighteen.

ISZE21PH must be installed in place of any E21 assembler exit as the two are mutually exclusive. It is recommended that the exit script is used and not the assembler exit as any future enhancements will be provided in the exit script only.

To install the E21 exit script for OLA, either copy source member ISZE21PH from the supplied .SISZCONF library into the customer ASCRIPT library, .SISZCASC, or copy source member ISZE21PH from the supplied .SISZCONF library into the customer library .SISZCCNF, then insert a COPY statement for ISZE21PH in each ISZCONxx member, and add the COPY ISZE21PH after the COPY ISZCOMON at the end of the ISZCONxx member in the .SISZCCNF library. (See also the *Online and Batch Administration* manual.)

To install the E21 exit script for Classic, copy source member ISZE21PH from the supplied .SISZCONF library into the customer library .SISZCUST. Then insert a COPY statement for ISZE21PH in each ISZCONxx member, and add the COPY ISZE21PH at the end of the ISZCONxx member in the .SISZCUST library.

Assembler E21 exit

Call information **General call information (pointed to by address 1)**

Signon validation call indicator 'E21'.

Reason code 'SIGINVAL'.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log and screen.

Terminal session information (pointed to by address 4)

Position	Len	Description	Exit script variables
1 - 4	4	Terminal task type	ec21_ttype
5 - 12	8	Terminal name	ec21_term
13 - 16	4	Terminal session id	
17 - 24	8	Logmode name (VTAM only)	ec21_logm
25 - 51	26	Bind image (VTAM only)	ec21_bind

User supplied user information (pointed to by address 5)

Position	Len	Description	Exit script variables
1 - 8	8	Entered user name	ec21_euser
9 - 16	8	Entered password	ec21_epass
17 - 24	8	Entered new password	ec21_enpass
25 - 32	8	Entered new profile	ec21_eprof
33 - 40	8	Entered new logon node	ec21_enode

Configuration data for the user (pointed to by address 6)

Position	Len	Description	Exit script variables
1 - 1	1	User definition in configuration, 'Y' or 'N'	ec21_config
2 - 2	1	Reserved	
3 - 4	2	Configuration authority	en21_cauth

Position	Len	Description	Exit script variables
5 - 12	8	Configuration password	ec21_cpass
13 - 20	8	Configuration profile	ec21_cprof
21 - 28	8	Configuration affinity	ec21_cnode

Signon parameter information, may be modified by the exit (pointed to by address 7)

Position	Len	Description	Exit script variables
1 - 8	8	New password	ec21_npass
9 - 16	8	New profile	ec21_nprof
17 - 18	2	New authority	en21_nauth

The following flags control whether the originally entered values are redisplayed when the return code is 4. The first only of these flags is also applicable to return code 0, when it is used for the multiple user facility.

Position	Len	Description	Exit script variables
19 - 19	1	Return code 0: Multiple users flag, 'M' or blank Return code 4: Redisplay user value flag, 'Y' or 'N'	ec21_ruser
20 - 20	1	Redisplay password flag, 'Y' or 'N'	ec21_rpass
21 - 21	1	Redisplay new password flag, 'Y' or 'N'	ec21_rnpass
22 - 22	1	Redisplay new profile name flag, 'Y' or 'N'	ec21_rnprof
23 - 30	8	New node/affinity	ec21_nnode
31 - 34	4	Address of logon data	ec21_ndata
35 - 35	1	Length of logon data	

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Signon is successful.
- 4 = Signon is unsuccessful, redisplay signon panel.
- 8 = Signon is unsuccessful, disconnect terminal.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.

44 = Reset all previous return codes for all exit points.

For any other value, 4 is assumed.

Signon completion exit point (E22)

This exit point is invoked when the user has finally been identified and the signon, or reconnection, has been accepted. The autoselect session detail number may be modified by the exit. The E22 exit might also be used to record the fact that the signon has been accepted. A message may be sent to the log and to the terminal.

Note This exit point is used to provide the ‘dynamic menus’ facility. For details, see ‘Implementing Dynamic Menus’ on page 306.

If the E22 exit point is used in conjunction with the signoff exit point (E29), a table or file may be maintained recording the signon and signoff times of each user. This may then be used for accounting purposes.

Call information **General call information (pointed to by address 1)**

Signon complete call indicator ‘E22’.

Reason code ‘SIGNON ’ or ‘RECON ’.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log and screen.

Terminal session information (pointed to by address 4)

Position	Len	Description	Exit script variables
1 - 4	4	Terminal task type	ec22_ttype
5 - 12	8	Terminal name	ec22_term
13 - 16	4	Terminal session id	
17 - 24	8	Logmode name (VTAM only)	ec22_logm
25 - 51	26	Bind image (VTAM only)	ec22_bind

Final user information (pointed to by address 5)

Position	Len	Description	Exit script variables	Note
1 - 8	8	User name	ec22_user	
9 - 16	8	Password	ec22_pass	
17 - 24	8	Profile	ec22_prof	
25 - 26	2	Authority	en22_auth	1

Note

1 User modifiable.

User options (pointed to by address 6, modifiable by the exit)

Position	Len	Description	Exit script variable
1 - 2	2	Autoselect session detail number	en22_autos

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Signon successfully completed.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

Input 3270 datastream exit point (E25)

The characteristics of the input/output exits differ from the other user exit points in several important respects. For details, see 'Input and output 3270 datastream exit points' on page 252.

IDLEDISC/IDLELOGOFF/IDLELOCK timer expiry exit point (E26)

This exit point is invoked when either the IDLEDISC, IDLELOGOFF or IDLELOCK timer has expired. The purpose of this exit is to determine whether the timeout is to be honoured, or an extension is to be granted to allow a further period of inactivity. This exit point is exit script only. The OUTSCAN TPSL verb may be used to examine the screen buffer (if available).

Call information **General call information**

Signon complete call indicator 'E26' (ec_rcode).

Reason code 'IDLEDISC' or 'IDLELOG' or 'IDLELOCK'(ec_reason).

For other general call information, see page 206.

Variable Access routine

Not applicable – exit script only.

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Honour timeout.
- 4 = Extend timeout period.
- 20= No application screen.
- 28 = The User exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

Signoff exit point (E29)

This exit point is invoked when the DISCONNECT or LOGOFF command is used to disconnect from Session Manager. It is also invoked when a user is disconnected or logged off from Session Manager due to expiry of the idle time interval, and also when Session Manager encounters an irrecoverable terminal error. As with the Signon exit point, a message may be sent to the log. The signoff event may be recorded in a table or file, and if the signon time has also been recorded, the interval may be used for accounting purposes.

Call information **General call information (pointed to by address 1)**

Signoff call indicator 'E29'.

Reason code:

'LOGOFF'	LOGOFF command.
'DISC'	DISCONNECT command.
'IDLEDISC'	IDLE disconnect.
'IDLELOG'	IDLE logoff.
'TERMDISC'	TERMERROR disconnect.
'TERMLOG'	TERMERROR logoff.
'STOPALL'	STOP ... ALL, FORCE, CLOSEDOWN.
'STOPDISC'	STOP ... (not stop all).

for other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log only.

Terminal session information (pointed to by address 4)

Position	Len	Description	Exit script variables
1 - 4	4	Terminal task type	ec29_ttype
5 - 12	8	Terminal name	ec29_term
13 - 16	4	Terminal session id	
17 - 24	8	Logmode name (VTAM only)	ec29_logm
25 - 51	26	Bind image (VTAM only)	ec29_bind

Final user information (pointed to by address 5)

Position	Len	Description	Exit script variables
1 - 8	8	User name	ec29_user
9 - 16	8	Password	ec29_pass
17 - 24	8	Profile	ec29_prof
25 - 26	2	Authority	en29_auth

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Signoff successfully completed.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

Slave session pre-initiation exit point (E31)

This exit point is called immediately prior to the initiation of an application session, that is when a user selects an application from a Menu panel. The main purpose of this exit point is to enable modification of some of the session parameters. Session start may also be recorded.

Call information **General call information (pointed to by address 1)**

Session initiation call indicator 'E31'.

Reason code 'SESSINIT'.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log and screen.

Terminal session information (pointed to by address 4)

Position	Len	Description	Exit script variables
1 - 2	2	Terminal task type	ec31_ttype
5 - 12	8	Terminal name	ec31_term
13 - 16	4	Terminal session id	
17 - 24	8	Logmode name (VTAM only)	ec31_logm
25 - 51	26	Bind image (VTAM only)	ec31_bind

Final user information (pointed to by address 5)

Position	Len	Description	Exit script variables
1 - 8	8	User name	ec31_user
9 - 16	8	Password	ec31_pass
17 - 24	8	Profile	ec31_prof
25 - 26	2	Authority	en31_auth
27 - 32	6	Spare	

Session information (pointed to by address 6)

May be modified by the exit, (except task identifier and username).

Position	Len	Description	Exit script variables
1 - 8	8	Task identifier	ec31_taskida
9 - 16	8	Username	ec31_taskidb
17 - 24	8	Applid (VTAM)	ec31_appl
25 - 32	8	ACB name (VTAM)	ec31_acb
33 - 40	8	Logmode name (VTAM)	ec31_slogm
41 - 48	8	Start script name	ec31_script
49 - 304	256	Logon data (VTAM only)	ec31_data

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Permit session to be initiated.
- 4 = Prevent session initiation.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

For any other value, 4 is assumed.

Slave session post-initiation exit point (E33)

This exit point is called when a session has been initiated. This enables response time monitors, such as NETSPY and ETE, to collect more comprehensive statistics. Also see exit E79, on page 242 and 'User audit facility' on page 284.

Call information **General call information (pointed to by address 1)**

Startup call indicator 'E33'.

Call reason 'SESSTART'.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log only.

Terminal session information (pointed to by address 4)

Position	Len	Description	Exit script variables
1 - 4	4	Terminal task type	ec33_ttype
5 - 12	8	Terminal name	ec33_term
13 - 16	4	Terminal session id	
17 - 24	8	Logmode name (VTAM only)	ec33_logm
25 - 51	26	Bind image (VTAM only)	ec33_bind

Final user information (pointed to by address 5)

Position	Len	Description	Exit script variables
1 - 8	8	User name	ec33_user
9 - 16	8	Password	ec33_pass
17 - 24	8	Profile	ec33_prof
25 - 26	2	Authority	en33_auth
27 - 32	6	Spare	

Session information (pointed to by address 6)

Position	Len	Description	Exit script variables
1 - 8	8	Session task id, SAAnnnnn, where nnnnn is the unique session number	ec33_taskida
9 - 16	8	Session task id, either userid or terminal name	ec33_taskidb
17 - 24	8	Applid (VTAM only)	ec33_applid
25 - 32	8	ACB (VTAM only)	ec33_acb
33 - 36	4	Session CID (VTAM only)	ec33_appl

Terminal data stream (pointed to by address 7)

Position	Len	Description	Exit script variables
1 - 2	1	Length of data stream to be sent the terminal. A value of 0, means the data stream will not be sent. Any other value, must equal the length of the data stream	
3 - 258	256	Terminal data stream. The maximum length is 256 bytes	ec33_dstream

Note The data stream is sent directly to the terminal, bypassing Session Manager data stream compression techniques.

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Signon successfully completed.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

Output 3270 datastream exit point (E35)

The characteristics of the input/output exits differ from the other user exit points in several important respects. For details, see 'Input and output 3270 datastream exit points' on page 252.

SIDLTIME timer expiry exit point (E36)

This exit point is invoked when the SIDLTIME timer has expired. The purpose of this exit is to determine whether the timeout is to be honoured, or an extension is to be granted to allow a further period of inactivity. This exit point is exit script only. The OUTSCAN TPSL verb may be used to examine the screen buffer (if available).

Call information **General call information**

Signon complete call indicator 'E36' (ec_rcode).

Reason code 'SIDLTIME' (ec_reason).

For other general call information, see page 206.

Variable Access routine

Not applicable – exit script only.

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Honour timeout.
- 4 = Extend timeout period.
- 20 = No application screen.
- 28 = The User exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

Slave session termination and SMF record exit point (E39)

This exit point is invoked immediately prior to session termination. Termination may be due to a variety of circumstances, for example, the application requested session termination, the user logged off, or as a result of commands such as RESET. The main function of this exit point is to record the fact that an application session has ended.

A sample E39 exit (ISZE39SM) is supplied with the product. This sample exit may be tailored to produce SMF records containing data to meet your installation requirements.

See also 'User audit facility' on page 284.

Call information **General call information (pointed to by address 1)**

Startup call indicator 'E39'.

Reason code 'SESSTERM'.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log and screen.

Terminal session information (pointed to by address 4)

Position	Len	Description	Exit script variables
1 - 4	4	Terminal task type	ec39_ttype
5 - 12	8	Terminal name	ec39_term
13 - 16	4	Terminal session id	
17 - 24	8	Logmode name (VTAM only)	ec39_logm
25 - 51	26	Bind image (VTAM only)	ec39_bind

Final user information (pointed to by address 5)

Position	Len	Description	Exit script variables
1 - 8	8	User name	ec39_user
9 - 16	8	Password	ec39_pass
17 - 24	8	Profile	ec39_prof
25 - 26	2	Authority	en39_auth

Position	Len	Description	Exit script variables
27 - 32	6	Spare	

Session information (pointed to by address 6)

Position	Len	Description	Exit script variables	Note
1 - 8	8	Task identifier	ec39_taskida	
9 - 16	8	Username	ec39_taskidb	
17 - 24	8	Applid (VTAM))	ec39_applid	
25 - 32	8	ACB name (VTAM))	ec39_acb	
33 - 40	8	Logmode name (VTAM))	ec39_logm	
41 - 44	4	Address of session statistics block		1

Note

- 1 Only available if STATS ON is specified for the session, else 0.

Session statistics block (pointed to by address in session info)

Only available if STATS ON is set for the session.

Position	Len	Description	Exit script variables	Note
1 - 4	4	'SSTB'		
5 - 8	4	'632'		
9 - 12	4	Session input: count	s_stat_sict	
13 - 20	8	Bytes	s_stat_siby	
21 - 24	4	Session output: count	s_stat_soct	
25 - 32	8	Bytes	s_stat_soby	
33 - 36	4	Outbound data: count	s_stat_ioct	
37 - 44	8	Input to MISER	s_stat_ioin	
45 - 52	8	Output from MISER	s_stat_ioon	
53 - 56	4	Inbound data: count	s_stat_iict	
57 - 64	8	Input to MISER	s_stat_iiin	
65 - 72	8	Output from MISER	s_stat_iion	
73 - 76	4	MISER read: count	s_stat_ifct	1
77 - 84	8	Bytes	s_stat_ifon	
85 - 88	4	Session Manager read: count	s_stat_ibct	2
89 - 96	4	Bytes	s_stat_ibon	

Position	Len	Description	Exit script variables	Note
97 - 100	4	No. of data streams compressed	s_stat_cmct	
101 - 108	8	Compression: no. of bytes in	s_stat_cmin	
109 - 116	8	No. of bytes out	s_stat_cmon	

Note

- 1 Application read buffers emulated by MISER.
- 2 Session Manager internal read buffers.

Response time measurement (RTM) information

Position	Len	Description	Exit script variables
117 - 120	4	RTM threshold 1, in milliseconds	s_rtm_rtmt1
121 - 124	4	RTM threshold 2, in milliseconds	s_rtm_rtmt2
125 - 132	8	TOD of last input from application 0 = no input outstanding	
133 - 140	8	TOD of last output from network 0 = no output outstanding	
141 - 148	8	Session start time, <i>hh:mm:ss</i>	s_rtm_strtt
149 - 156	8	Session start date, default format <i>mm/dd/yy</i>	s_rtm_strtd
157 - 164	8	Time RTM statistics were last reset, <i>hh:mm:ss</i>	s_rtm_restd
165 - 172	8	Date RTM statistics were last reset, default format <i>mm/dd/yy</i>	s_rtm_restd
173 - 180	8	Userid of user who last reset the RTM statistics	s_rtm_restu
181 - 188	8	Terminal id where the RTM statistics were last reset	s_rtm_restm
189 - 196	8	Terminal type where the RTM statistics were last reset	s_rtm_restp
197 - 200	4	Spare	

Total response table

Position	Len	Description	Exit script variables
201 - 202	2	Offset to latest Total response time	
204 - 204	2	Spare	

Position	Len	Description	Exit script variables
205 - 208	4	No. of Total response times < threshold 1	s_rtm_tot1
209 - 212	4	No. of Total response times > threshold 1 but < threshold 2	s_rtm_tot2
213 - 216	4	No. of Total response times > threshold 2	s_rtm_tot3
217 - 224	8	Total of all Total response times, in milliseconds (a floating point number) <i>average</i> Total response time is available in variable s_rtm_totav.	s_rtm_totav
225 - 228	4	These five fullwords contain the last five Total response times, in milliseconds. These are entered in rotation, with the latest response found at the position which is 225 plus the offset stored at position 201. The penultimate response is then at 225+(offset-4), unless offset is zero when the penultimate is at position 241.	
229 - 232	4		
233 - 236	4		
237 - 240	4		
241 - 244	4		
		The following five fullwords contain the five longest Total response times, in milliseconds. The last fullword contains the longest, the last but one the next longest, and so on.	
245 - 248	4	Fifth longest Total response time	
249 - 252	4	Fourth longest Total response time	
253 - 256	4	Third longest Total response time	
257 - 260	4	Second longest Total response time	
261 - 264	4	Longest Total response time	s_rtm_totlg

Position	Len	Description	Exit script variables
		The five longest Total response times occurred at (in the same sequence):	
265 - 280	16	<i>hh:mm:ss mm/dd/yy</i> (default date format)	
281 - 296	16	<i>hh:mm:ss mm/dd/yy</i> (default date format)	
297 - 312	16	<i>hh:mm:ss mm/dd/yy</i> (default date format)	
313 - 328	16	<i>hh:mm:ss mm/dd/yy</i> (default date format)	
329 - 344	16	<i>hh:mm:ss mm/dd/yy</i> (default date format)	s_rtm_totIt s_rtm_totId

Network response table

Position	Len	Description	Exit script variables
345 - 346	2	Offset to latest Network response time	
347 - 348	2	Spare	
349 - 352	4	No. of Network response times < threshold 1	s_rtm_net1
353 - 356	4	No. of Network response times > threshold 1 but < threshold 2	s_rtm_net2
357 - 360	4	No. of Network response times > threshold 2	s_rtm_net3
361 - 368	8	Total of all Network response times, in milliseconds (a floating point number) <i>average</i> Network response time is available in variable s_rtm_netav	s_rtm_netav
369 - 372	4	These five fullwords contain the last five Network response times, in milliseconds. These are entered in rotation, with the latest response found at the position which is 369 plus the offset stored at position 345. The penultimate response is then at 369+(offset-4), unless offset is zero when the penultimate is at position 385.	
373 - 376	4		
377 - 380	4		
381 - 384	4		
385 - 388	4		
		The following five fullwords contain the five longest Network response times, in milliseconds. The last fullword contains the longest, the last but one the next longest, and so on.	
389 - 392	4	Fifth longest Network response time	
393 - 396	4	Fourth longest Network response time	
397 - 400	4	Third longest Network response time	
401 - 404	4	Second longest Network response time	
405 - 408	4	Longest Network response time	s_rtm_netlg

Position	Len	Description	Exit script variables
		The five longest Network response times occurred at (in the same sequence):	
409 - 424	16	<i>hh:mm:ss mm/dd/yy</i> (default date format)	
425 - 440	16	<i>hh:mm:ss mm/dd/yy</i> (default date format)	
441 - 456	16	<i>hh:mm:ss mm/dd/yy</i> (default date format)	
457 - 472	16	<i>hh:mm:ss mm/dd/yy</i> (default date format)	
473 - 488	16	<i>hh:mm:ss mm/dd/yy</i> (default date format)	s_rtm_netlt s_rtm_netld

Application response table

Position	Len	Description	Exit script variables
489 - 490	2	Offset to latest application response time	
491 - 492	2	Spare	
493 - 496	4	No. of application response times < threshold 1	s_rtm_app1
497 - 500	4	No. of application response times > threshold 1 but < threshold 2	s_rtm_app2
501 - 504	4	No. of application response times > threshold 2	s_rtm_app3
505 - 512	8	Total of all application response times, in milliseconds (a floating point number) <i>average</i> application response time is available in variable s_rtm_appav.	s_rtm_appav
513 - 516	4	These five fullwords contain the last five application response times, in milliseconds. These are entered in rotation, with the latest response found at the position which is 513 plus the offset stored at position 489. The penultimate response is then at 513+(offset-4), unless offset is zero when the penultimate is at position 529.	
517 - 520	4		
521 - 524	4		
525 - 528	4		
529 - 532	4		

Position	Len	Description	Exit script variables
		The following five fullwords contain the five longest application response times, in milliseconds. The last fullword contains the longest, the last but one the next longest, and so on.	
533 - 536	4	Fifth longest application response time	
537 - 540	4	Fourth longest application response time	
541 - 544	4	Third longest application response time	
545 - 548	4	Second longest application response time	
549 - 552	4	Longest application response time	s_rtm_applg
		The five longest application response times occurred at (in the same sequence):	
553 - 568	16	hh:mm:ss mm/dd/yy (default date format)	
569 - 584	16	hh:mm:ss mm/dd/yy (default date format)	
585 - 600	16	hh:mm:ss mm/dd/yy (default date format)	
601 - 616	16	hh:mm:ss mm/dd/yy (default date format)	
617 - 632	16	hh:mm:ss mm/dd/yy (default date format)	s_rtm_applt s_rtm_appld

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = The exit point has successfully completed.
- 4 = The session will be restarted unless RESET, END, CLOSEDOWN or STOP USER *seln-id* are entered, in which case it is ignored.
- 8 = Causes Session Manager to logoff the user. This means the user does not have to see the Menu panel if they do not wish to. Current LOGDISC settings are honoured.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

For any other value, 0 is assumed.

User exit replacement exit point (E71)

When an UPDATE EXIT command has been issued, this exit point is called immediately prior to the replacement of the old exit with the new exit. This enables the old exit to close files or perform any other necessary functions before the new exit comes into operation.

Call information **General call information (pointed to by address 1)**

Exit replacement call indicator 'E71'.

Reason code 'EXITUPDT'.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log only.

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = The exit point has successfully completed.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

For any other value, 0 is assumed.

Session switch exit point (E79)

This exit point is called when a Session Manager user switches between a session and a Session Manager function.

The function of this exit is to allow response time monitors, such as NETSPY and ETE, to collect more comprehensive statistics. This is achieved by informing the response time products when the Session Manager switches between applications. See also the exit E33 on page 229, and 'Producing statistics for NETSPY and ETE' on page 251.

Call information **General call information (pointed to by address 1)**

Exit update call indicator 'E79'.

Reason code 'VTAM', 'PASS' (VTAM only), or 'ISZSMGR':

VTAM VTAM application session

PASS Application has issued a CLSDST PASS to another applid (VTAM only)

ISZSMGR Session Manager function

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information (pointed to by address 3)

See page 207. Sends message to log only.

Terminal session information (pointed to by address 4)

Position	Len	Description	Exit script variables
1 - 4	4	Terminal task type	ec79_ttype
5 - 12	8	Terminal name	ec79_term
13 - 16	4	Terminal session id	
17 - 24	8	Logmode name (VTAM only)	ec79_logm
25 - 51	26	Bind image (VTAM only)	ec79_bind

User information (pointed to by address 5)

Position	Len	Description	Exit script variables
1 - 8	8	User name	ec79_user
9 - 16	8	Password	ec79_pass
17 - 24	8	Profile	ec79_prof
25 - 26	2	Authority	en79_auth
27 - 32	6	Spare	

Session information (pointed to by address 6)

Position	Len	Description	Exit script variables
1 - 8	8	Session task id - xxxnnnnn where xxx is the function id, and nnnnn is the unique task number.	ec79_taskida
9 - 16	8	Session task id, either userid or terminal name.	ec79_taskidb
17 - 24	8	Applid (VTAM)	ec79_applid
25 - 32	8	ACB name (VTAM)	ec79_acb
		TID=nnnn (internal function) where nnnn is the task type of the internal function that is being made visible. Possible values for nnnn are: 28 Menu 44 CUT 48 PASTE 52 SIGNON 58 QUERY 60 VIEW 64 DLOG 68 STORAGE display 72 DTERM 76 GFS display 88 HELP 96 BROADCAST 100 SPYTELL 104 IDLETIME 112 DEMO DISPLAY 152 SHARESESS secondary	
33 - 36	4	Session CID (VTAM only)	
37 - 44	8	Session Manager APPL name	

Position	Len	Description	Exit script variables
45 - 52	8	(VTAM only) Old application name for a PASS call; otherwise the current/latest APPL name if a CLSDST PASS has been issued by the application	ec79_oapplid
53 - 60	8	(VTAM only) New application name (Blank if call is not PASS)	ec79_napplid

Terminal data stream (pointed to by address 7)

Position	Len	Description	Exit script variables
1 - 2	1	Length of data stream to be sent the terminal. A value of 0, means the data stream will not be sent. Any other value, must equal the length of the data stream.	
3 - 258	256	Terminal data stream. The maximum length is 256 bytes.	ec79_dstream

Note The data stream is sent directly to the terminal bypassing Session Manager data stream compression techniques.

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = Signon successfully completed.
- 28 = The user exit has no processing for this particular call.
- 40 = Suppress further calls which have the same call indicator.
- 44 = Reset all previous return codes for all exit points.

Closedown exit point (E99)

This exit point is invoked at Session Manager closedown after all sessions and terminals are closed. The primary functions of this exit point are to enable the shutdown to be noted, and to provide an opportunity to perform any tidying operations.

Call information **General call information (pointed to by address 1)**

Closedown call indicator 'E99'.

Reason code 'SHUTDOWN'.

For other general call information, see page 206.

Variable Access routine (pointed to by address 2)

See page 206.

Output message information, may be modified by the exit (pointed to by address 3)

See page 207. Sends message to log only.

Return codes

When control is returned to Session Manager, Register 15 must contain a return code. This should be set as follows:

- 0 = The exit point has successfully completed.
- 28 = The user exit does not support this function.

For any other value, 0 is assumed. Unlike most of the exit points return codes 40 and 44 are irrelevant, since the exit point is called only once.

Variable access

Each exit point's parameter list enables the exit to directly inspect, and possibly modify, various system and user-defined variables applicable to that particular exit point. However, if any other existing applicable variables need to be viewed or amended, or if new variables are to be defined, the user exit may call the 'Variable Access routine'. The address of this routine is provided in the parameter list.

The routine may be called with one of two functions, either GET to obtain the current values of variables, or PUT to change the value of existing variables and define new variables.

Any number of variables can be accessed in a single call to the Variable Access routine. This is much more efficient than calling the routine for each variable.

Note, however, that not every type of variable is accessible from each exit point, and terminal details are not set up when the user is disconnected.

The following table shows which variables may be accessed:

Exit	Description	Global	User	Local	Subscribed session	Current session
E01	Initialization	Y				
E05	Configuration statement processing	Y	ISZSMGR			
E06	Application Status change	Y	ISZSMGR			
E08	Timer Interval	Y	ISZSMGR			
E09	Script Call	Y	Y	Y	If in Windows	Y
E11	Logon	Y	Y	Y	Y	
E21	Signon Validation	Y	Y	Y	Y	
E22	Signon Completion	Y	Y	Y	Y	
E29	Signoff	Y	Y		Y	
E31	Slave session pre-initiation	Y	Y	Y	Y	Y
E33	Slave session post-initiation	Y	Y	Y	Y	Y
E39	Slave session termination	Y	Y	Y	Y	Y
E71	User Exit replacement	Y	ISZSMGR			
E79	Session switch	Y	Y		Y	
E99	Closedown	Y	ISZSMGR			

Use of user-defined variables

All 'gc_' and 'gn_' variables are available in all Exit scripts.

In Exit scripts E11, E21, E22, E29, E31, E33, E39 and E79, the user-associated 't_', 'uc_' and 'un_' variables are available and related to the user or terminal which is the subject of the Exit script. Local user-defined variables 'lc_' and 'ln_' are available, but are defined and available for the length of the function only. For example:

- The 'local variable logon/signon function set' is available to the E11, E21 and E22 exits as well as the Signon panel.
- The 'local variable session set' is available to the E31, E33 and E39 exits as well as the sessions Session scripts, USERMSG, SES1 and SES2 panels.

's_' variables are available without a subscript in the session related Exit scripts (E31, E33, E39) where they refer to the session that is the subject of the exit. These session variables can be referenced from the user related exits (E11, E21, E22, E29 and E79) via a subscript. The session exits can also refer to a user's other session variables via a subscript. The system related Exit scripts (E05, E06, E08 and E99) cannot access any session variables

The system related Exit scripts (E05, E06, E08, E71 and E99) are associated with a system user, that is, t_user = ISZSMGR, t_term = SYSTEM, whose user related t_ variables and user-defined uc_ and un_ variables are available. The system user is not shown in QUERY displays and cannot be sent a message or broadcast.

Linkage conventions

Linkage to the Variable Access routine from the user exit is provided by the following statements:

LA	0,x	Where x is 0 for GET, x is 4 for PUT
LA	1,PARMBLOK	Point to parameter block, see below
L	15,VARRTN	Variable Access routine address, provided in the parameter list
BALR	14,15	Call the routine

Input to the routine

Input to the routine consists of a parameter block pointed to by Register 1 and a call indicator in Register 0.

Variable Access Parameter Block

This consists of one or more variable value store addresses, and should be terminated by a 4-byte zero end of parameter block indicator. The following shows an example of a parameter block:

```
PARMBLOK DS 0F          *** Start of parm block ***
PA1ADDR  DC A(USERID)   Points to userid variable value store
PA2ADDR  DC A(OPASWD)   Points to password variable value store
PA3ADDR  DC A(NPASWD)   Points to new password variable value store
PA4END   DC A(0)        *** End of parm block ***
```

Variable Value Store Area

This is the format of the Variable Value Store Area:

Position	Len	Description
1 - 16	16	Variable name
17 - 17	1	Access return code for the variable in hex
18 - 18	1	Length in hex of data in field after a GET request. This is a read-only field set by Session Manager.
19 - 19	1	This is set by the user exit. For a GET request it gives the length of the field provided to receive the data. For a PUT request, it gives the length of the variable data.
20	1-255	Field provided to receive data

Examples:

- The following is an example of a value store area set up for a GET operation:

```
USERID    DC CL16'T_USER'   Variable name
USERIDRC  DC AL1(0)        Access return code in hex
USERIDDL  DC AL1(0)        Length of data in field
USERIDFL  DC AL1(L'USERIDF) Length of field provided
USERIDF   DS CL8           Field provided to receive data
```

The Variable Access routine stores the actual length in USERIDDL, and the value in USERIDF.

- The following is an example of a value store area set up for a PUT operation for a character string:

```
GCWARN    DC CL16'GC_WARNING' Variable name
GCWARNRC  DC AL1(0)        Access return code in hex
GCWARNDL  DC AL1(26)       Length of data in field
GCWARNFL  DC AL1(255)     Length of field provided
GCWARNF   DC CL255'NORMAL  Field containing data
           CLOSEDOWN AT 1830'
```

3 The following is an example of a value store area set up for a PUT operation for a 2 byte numeric variable:

GNNUMBER	DC	CL16 'GN_NUMBER'	Variable name
GNNUMRC	DC	AL1(0)	Access return code in hex
GNNUMDL	DC	AL1(L'GNNUMF)	Length of data in field
GNNUMFL	DC	AL1(L'GNNUMF)	Length of field provided. This must be the same as the data length when using numeric variables.
GNNUMF	DC	AL2(55)	Unsigned hex 2 or 4 byte field

Output from the routine

For the GET function, the value store areas are updated with variable values and actual lengths.

Register contents for a GET function

R15	0	4	8	12	16	32	36
R0	0	address	address	address	address	-	-

Register contents for a PUT function

R15	0	4	8	12	16	20	24	32	36
R0	0	0	address	address	address	address	address	-	-

36 in R15 indicates R0 on entry to the routine was neither 0 nor 4. Otherwise, R15 contains the highest Variable Access return code and R0 contains the address of that Variable Value Store Area.

On return from the routine an access return code is placed in the store area for each variable. The return codes are as follows:

Hexadecimal	Decimal	Description
X'00'	00	The required operation was performed successfully, that is, for a GET, the variable was retrieved, or a PUT, the variable was updated
X'04'	04	For a GET, the variable has not yet been defined. For a PUT, a new variable has been created.
X'08'	08	The variable name is invalid. Either the Session Manager supplied variable name is not recognizable, or the user-defined variable name has an invalid prefix. See 'Session Manager variables' in the <i>Panels, Scripts and Variables</i> manual for specific information.
X'0C'	12	The variable is outside the scope of the exit at this point and is therefore not accessible. For example, when the E01 exit tries to access a user or local variable.
X'10'	16	A subscript is needed, but is either missing or it is too large. For example, an attempt to access an unsubscripted Session Manager session variable in the E22 (Signon Completion) exit point would cause this return code since there is no current session at that time.
X'14'	20	This indicates that a numeric variable has a length other than 2 or 4.
X'18'	24	For PUT operations only, the variable is not updateable. For example, t_date.
X'20'	32	There has been an abend when attempting to access or update the variable, and the exit should complete immediately.

Producing statistics for NETSPY and ETE

The ISZE33NY, ISZE79NY and ISZE79ET user exits (exit points E33 and E79) assist the products NETSPY and ETE in detecting slave session initiation and session switching. This enables NETSPY, ETE, NPM or VITALSIGNS to produce more accurate statistics for response time measurement and traffic flow.

The ISZE33NY exit is used only as a front-end to call the ISZE79NY exit, which is where all the processing is done. (A slave session termination exit (E39) is not needed because both the NETSPY and ETE products can detect when a slave session ends.)

The ISZE79NY exit uses control variables to store details about the session switch so that it can determine which application and CID are being switched. The APPL name being switched from and the CID for that session are inserted in a data stream which both NETSPY and ETE can intercept.

Input and output 3270 datastream exit points

See

- ‘Overview’ below
- ‘Characteristics of the input/output exits’ below
- ‘Storage Obtain/Free routine’ on page 258
- ‘3270 datastream areas’ on page 259
- ‘Variable Access routine’ on page 259
- ‘Issuing Session Manager commands’ on page 260
- ‘Issuing the HARDCOPY command’ on page 261
- ‘Setting up an output message’ on page 261
- ‘Sample exit’ on page 261
- ‘Sample macros’ on page 261

See also ‘User audit facility’ on page 284.

Overview

The input and output 3270 datastream exits (hereafter “input/output exits”) are discussed separately, since their characteristics differ from the other user exit points in several important respects. For details, see ‘Characteristics of the input/output exits’ below.

Exit		When invoked	See
E25	(Input 3270 datastream exit)	Upon receipt of a terminal input 3270 datastream, prior to Session Manager escape sequence detection and processing	page 272
E35	(Output 3270 datastream exit)	Upon receipt of an application output 3270 datastream, prior to session script processing	page 231

Characteristics of the input/output exits

This section describes the characteristics of the E25 (input 3270 datastream) and E35 (output 3270 datastream) user exit points.

See

- ‘Configuration’ on page 253
- ‘General’ on page 253
- ‘Environmental’ on page 253
- ‘Common exit point facilities’ on page 253
- ‘Common exit conventions’ on page 254
- ‘Register conventions’ on page 254
- ‘Entry Parameter List format’ on page 254

Configuration

Whereas all other exits are specified on the `OPTION` statement, the load module names for the E25 and E35 exits are identified on the `SYSTEM` statement by the `INPUTEXIT` and `OUTPUTEXIT` parameters respectively. An additional, related, `SYSTEM` statement parameter, `EXITWALEN`, specifies the size of the work area made available to the exits. For details on these parameters, see the *Technical Reference*.

These exit points are not refreshable upon dynamic update of the `SYSTEM` configuration statement.

General

The input/output exits can not run as scripts.

A unique load module must be created for each of the input/output exits points, and they can not be managed using the Multiple Exit Driver facility.

If identified on the `SYSTEM` statement at startup, these exits are loaded automatically as part of system initialization, and are deleted at shutdown. If not specified at startup, an Update of the configuration will not reload these exits even if the `SYSTEM` statement has been changed in respect of these exit-related parameters.

Environmental

For the standard user exit points, code is scheduled to run under a separate TCB, different to that of the Session Manager job step task. For performance reasons, the E25 and E35 exit code is given control directly by Session Manager. This places some restrictions on what the exit point may do, in order to keep the path length of the code as short as possible and therefore minimize any impact to Session Manager users. The exit point code should not:

- Issue an operating system `WAIT`, or use any service that implies a wait (for example, I/O).
- Issue SVCs or use operating system macros that cause SVCs to be issued.
- Issue PC calls, or use operating system macros that cause PC calls to be issued.

The exit points must be linked `AMODE(31) RMODE(ANY)`, and run in 31-bit mode.

Also, the exit points must be reentrant, and make use of the exit work area and/or Session Manager storage obtain/release facility provided, where necessary.

The exit point load modules must reside in a library identified in the Session Manager startup `JCL STEPLIB/JOBLIB` concatenation or `LINKLIST`. All libraries in the `STEPLIB` concatenation must be APF-authorized.

Common exit point facilities

The E25 and E35 exit points can:

- Inspect, add, modify, or delete user-defined variables.
- Inspect or modify certain Session Manager variables.
- Prepare a single message for output by Session Manager to the log (message numbers 0600 to 0699 are available for the exits).
- Write direct to the console, depending on `MESSAGE LOG` parameter setting.
- Schedule execution of a session script if one is not already running.
- Obtain and release Session Manager storage.
- Issue Session Manager commands.

- Create a hardcopy of the screen for current application session; cause Session Manager to start/stop hardcopying screens for an application session.

Note When hardcopying screens for sessions that do not have MISER active, the overhead of an additional read buffer to capture the screen image will be incurred.

Common exit conventions

Standard assembler subroutine linkage conventions should be used. To ensure re-entrance and avoid the use of SVC/PC calls, Session Manager supplies the address of a 72-byte save area as one of the parameters on entry. The sample macros, ISZEENT (see page 262) and ISZERET (see page 263), can be used to establish entry to the code, and exit to Session Manager.

Register conventions

On entry to the exit point, the register contents are set as follows:

Register	Contents
R1	Address of the address of the Entry Parameter List (EPL)
R13	Caller's (Session Manager's) register save area address
R14	Return address
R15	Entry point address
R0,R2-R12	Zero

R15 should normally be set to 0 on return to Session Manager; if set to 40 (decimal) then Session Manager will disable all further calls to the exit point until the system is restarted. This may be useful, for instance, where the exit code detects that the Exit Work Area is of insufficient size to allow correct execution.

Entry Parameter List format

The Entry Parameter List (EPL) consists of a list of addresses as follows:

Address	Description
1	Address of the General Call Information area
2	Address of the Session Manager Variable Access routine
3	Address of the Output Message Area
4	Address of exit Input Parameter Block
5	Address of exit Return Code/Action Block
6	Address of the Session Manager obtain/free storage routine
7	Address of the Session Manager command-issuing routine
8	Address of 72-byte register save area for use by the exit
9-24	Reserved for future use

This area can be mapped using sample macro ISZEPLST (see page 267).

Address 1 – General call information area

Position	Len	Description	Information
1-3	3	Call indicator 'Enn'	'E25' or 'E35'
4-4	1	Spare	
5-12	8	Reason code 'reason'	'3270IN' or '3270OUT'

Position	Len	Description	Information
13-20	8	Date in either <i>dd/mm/yy</i> or <i>mm/dd/yy</i> format	
21-24	4	Julian date in packed decimal (<i>00yydddF</i>)	
25-32	8	Time in <i>hh:mm:ss</i> format	
33-40	8	TOD clock value	
41-44	4	Address of statistics block	
45-112	66	Reserved for future use	

This area can be mapped using sample macro ISZEGCIA (see page 271).

Address 2 – Session Manager Variable Access routine

Address 2 points to the Variable Access routine. Sample macro ISZEVAR (see page 265) may be used to invoke this routine.

Address 3 – Output Message Area

The information in this area may be modified by the user exit.

Position	Len	Description	Information
1-4	4	Log message number	0600-0699
5-8	4	Log message length	
9-140	132	Log message text	
141-144	4	Spare	
145-208	64	Reserved for future use	

This area can be mapped using sample macro ISZEOMA (see page 270).

Address 4 – Input Parameter Block

Contains data such as information contained within certain Session Manager variables, and so on, that is likely to be required at the exit point to aid decision processing.

Position	Len	Description	Information
1-4	4	Address of the exit work area	
5-8	4	Length of the exit work area	As specified by the EXITWALEN parameter
9-12	4	Address of the 3270 datastream	
13-16	4	Length of the 3270 datastream	
17-17	1	Flag	Common to E25 and E35 exit points (see Note 1)

Position	Len	Description	Information
18-18	1	Flag	E25 exit point (see Note 2)
19-19	1	Flag	E35 exit point
20-20	1	Flag	Spare
21-28	8	Userid	
29-36	8	Terminal Id	
37-44	8	VTAM Logmode name	
45-52	8	APPLid	
53-60	8	VTAM ACB name	
61-64	4	Session number	
65-72	8	User task id	
73-80	8	Session task id	
81-84	4	Terminal task type	
85-110	26	VTAM bind image	
111-176	66	Reserved for future use	

Notes

- 1 X'80': Session script is running.
X'40': MISER is active for the application session.
- 2 X'80': Input from Application session.
X'40': Input from Session Manager function.

The Input Parameter Block can be mapped using sample macro ISZEIPB (see page 268).

Address 5 – Return Code/Action Block

The information placed in this area by the exit can be used to influence subsequent Session Manager processing.

Position	Len	Description	Information
1-4	4	Address of the replacement 3270 datastream	
5-8	4	Length of the replacement 3270 datastream	
9-9	1	Flag	Common to E25 and E35 exit points (see Note 1)
10-10	1	Flag	E25 exit point (see Note 2)
11-11	1	Flag	E35 exit point (see Note 3)
12-12	1	Flag	Spare

Position	Len	Description	Information
13-20	8	Script name	If flag in position 9 set to X'80'
21-24	4	Address of the 3270 datastream to be sent to the terminal	If flag in position 10 set to X'04'
25-28	4	Length of the 3270 datastream to be sent to the terminal	If flag in position 10 set to X'04'
29-36	8	HCPROFILE name	For a hardcopy request, override the user's hardcopy profile name
37-38	2	HCOPTION number	For a hardcopy request, override the user's hardcopy option no.
39-84	46	Reserved for future use	

Notes

- X'80': Schedule session script to be run.
X'40': Hardcopy application screen.
- X'80': Reject input datastream.
X'40': Unlock keyboard.
X'20': Trace current datastream.
X'10': Start user session tracing.
X'08': Stop user session tracing.
X'04': Send 3270 datastream to the terminal.
X'02': Start hardcopying application input screens.
X'01': Stop hardcopying application input screens.
- X'80': Reject output datastream.
X'40': E35 exit-supplied application input datastream.
X'02': Start hardcopying application output screens.
X'01': Stop hardcopying application output screens.

The Return Code/Action Block can be mapped using sample macro ISZERCAB (see page 269).

Address 6 – Session Manager Storage Obtain/Free routine

Address 6 points to the Session Manager Storage Obtain/Free routine. Sample macro ISZESTOR (see page 263) may be used to invoke this routine.

Address 7 – Session Manager command-issuing routine

Address 7 points to the Session Manager routine for issuing commands. Sample macro ISZECMD (see page 266) may be used to invoke this routine.

Address 8 – 72-byte register save area

Address 8 points to a 72-byte area to be used by the exit as a re-entrant area for standard assembler linkage. Sample macro ISZEENT (see page 262) uses this area.

Storage Obtain/Free routine

Usage

Note Use this routine only when it is absolutely necessary.

By specifying an appropriate value for the `EXITWALEN` parameter on the `SYSTEM` statement, the E25 or E35 exit point should have sufficient working storage made available to it *automatically*. If it is necessary to obtain further storage, it is usually the responsibility of the exit point code to free that storage; failure to do so may result in ongoing depletion of above-the-line virtual storage.

In these cases, however, it is not necessary for the exit point to free that storage:

- Where the E25 or E35 exit point is to supply an alternative 3270 datastream to be used instead of the one passed to the exit point.

In this case, the address of the area is returned to Session Manager in the `RABRDST@` field of the Return Code/Action Block area.

- Where the E35 exit point is to pass an exit-supplied 3270 datastream as input to an application.

In this case, the address of the area is returned to Session Manager in the `RABAIDS@` field of the Return Code/Action Block area.

Note In this document, the `0x7C` (that is, `x'7C'`) character is always presented as the `@` sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate `0x7C` character symbol for the code page you are using.

The exit point code should, if possible, use the supplied work area to contain the datastream. If the routine has been used to allocate additional storage to be used for a datastream, Session Manager will recognize that the address is not within the supplied exit work area, and will therefore be able to release the storage when it is no longer required *automatically*.

Invocation

To invoke the Storage Obtain/Free routine, supplied macro `ISZESTOR` (see page 263) can be used, and its use is recommended. Alternatively, the following registers and parameters need to be set up for the call:

R14	Return address
R15	Routine entry address
R1	Pointer to a 3-word parameter list in reentrant storage – for example, exit work area. Contents as follows:

Word 1: Address of the Exit Parameter List

Word 2: Address of area to be freed,
TYPE=FREE, or 0 for TYPE=OBTAIN

Word 3: Length of area to be obtained,
TYPE=OBTAIN, or 0 for TYPE=FREE

Register contents on return

The register contents on return are:

R1	Address of area successfully OBTAINED, otherwise unpredictable
----	--

R15	0 = Successful
	4 = Successful, length truncated
	8 = Unsuccessful
	-4 = Invalid request

The area returned upon successful execution of TYPE=OBTAIN requests is prefixed by a control area which, amongst other information, gives details of the total length of the area. For this reason, the TYPE=FREE request only requires the storage address, and not the length.

3270 datastream areas

If an exit needs to build a 3270 datastream to be passed back to Session Manager for subsequent processing then, if possible, it should be built within the exit work area supplied. The datastream may begin anywhere within the exit work area, as long as it will fit wholly within that area. The address of the first byte of the datastream should be passed back in the Return Code/Action Block, along with the datastream length. Session Manager will recognize, from the address, that the datastream is contained within the exit work area.

In the case of a *replacement* datastream, even if it is exactly the same length as the one supplied to the exit call, overwriting the original area in the exit point *is not recommended*, although there is no protection against this. If the original datastream *is modified in place by the exit* then the Return Code/Action Block *must not* specify a replacement datastream address or length value.

If an area larger than that available within the exit work area is required to build a datastream then additional storage can be obtained using the Storage Obtain/Free routine. In this case, the first byte of the datastream *must* start in the first byte of the obtained area. The address of the first byte *must* be passed back in the Return Code/Action Block, along with the length of the datastream built (which can be smaller than the obtained length but *must not be larger*).

Session Manager recognizes, *by the address and length of a datastream* supplied in the Return Code/Action Block, whether it resides within the exit work area or an exit-obtained area, and uses this information to free the storage automatically when it is no longer required.

Variable Access routine

Usage

Other than the call interface, the use of this routine is exactly the same as for the standard exit points.

Invocation

To invoke the Variable Access routine, supplied macro ISZEVAR (see page 265) can be used, and its use is recommended. Alternatively, the following registers and parameters need to be set up for the call:

R14	Return address
R15	Routine entry address

R1 Pointer to a 3-word parameter list in reentrant storage – for example, exit work area. Contents as follows:

Word 1: Address of the Exit Parameter List

Word 2: Address of the Variable Access Parameter Block

Word 3: Call indicator – 0 (GET) or 4 (PUT)

Format of Areas See ‘Variable Access Parameter Block’ on page 248 and ‘Variable Value Store Area’ on page 248 for the format of each respective area.

Register contents on return The register contents on return are:

R15 Highest Variable Access return code

R0 Pointer to first Variable Value Store Area in error, or 0

For more information, see ‘Output from the routine’ on page 249.

Variables accessible Variables accessible are as for E33 exit point.

Note Local variables (LC_ and LN_) are available to the E25 input exit only when the datastream originates from an application session, and not from a Session Manager function such as DLOG.

Issuing Session Manager commands

Invocation To invoke the Session Manager routine, supplied macro ISZECMD (see page 266) can be used, and its use is recommended. Alternatively, the following registers and parameters need to be set up for the call:

R14 Return address

R15 Routine entry address

R1 Pointer to a 2-word parameter list in reentrant storage – for example, exit work area. Contents as follows:

Word 1: Address of the Exit Parameter List

Word 2: Address of the Command Area

Commands will be issued with an AUTHORITY of 9.

Format of Command Area The format of the Command Area is:

Position	Len	Description
1-2	2	Length of command
3- <i>n</i>	<i>n</i>	Command string

Register contents on return The register contents on return are:

- R15 Contains one of these return codes after the command-issuing routine has run:
- 0 = Command issued successfully
 - 4 = Error (t_message will contain an error message and t_rc will be set to 4)
 - 8 = Command length is greater than 32767 characters

Issuing the HARDCOPY command

The HARDCOPY command may be issued directly from the E25/E35 exits, but its use is not recommended, and an alternative exit facility is provided for this purpose. If you do, however, decide to issue the HARDCOPY command directly from the E25/E35 exits then the following restrictions apply:

- The HARDCOPY command will not work if it is issued within the E35 exit, and MISER is not active for the application session.
- If the HARDCOPY command is issued from the E35 exit, and MISER is active, the actual screen hardcopied will not be the one generated by the datastream presented to the E35 exit. This is due to the positioning of the E35 exit point call in relation to the point at which MISER captures the datastream.
- If the HARDCOPY command is issued from the E25 exit, and MISER is not active for the application session, then an additional read buffer will be used to capture the screen image.

Setting up an output message

The Output Message Area is similar in format to the one used by the standard exit points only simpler, since the message can only be written to the Audit log (or console if the MESSAGE LOG options are set to do this). The way in which the Output Message Area fields are formatted/used is the same as for the standard exits.

Sample exit

A sample 3270 datastream input exit can be found in member ISZE25CM, which is supplied in library .SISZCONF. This sample exit enables a user to issue the Session Manager MSG command directly from an application session, rather than having to escape to the menu first. For example, in TSO you could type:

```
//MSG 'message-text' User user-name
```

and Session Manager would send the message to the nominated user.

Sample macros

These sample macros, which can be found in library .SISZCONF, are supplied to assist you with calling the service routines:

Macro	Usage	See
ISZEENT	Exit module code entry, standard assembler linkage	below
ISZERET	Exit module return to Session Manager, standard assembler linkage	page 263

Macro	Usage	See
ISZESTOR	Invoke Session Manager Storage Obtain/Free routine	page 263
ISZEVAR	Invoke Session Manager Variable Access routine	page 265
ISZECMD	Invoke Session Manager command-issuing routine	page 266
ISZEPLST	Map Exit Parameter List	page 267
ISZEIPB	Map exit Input Parameter Block	page 268
ISZERCAB	Map exit output Return Code/Action Block	page 269
ISZEOMA	Map exit Output Message Area	page 270
ISZEGCIA	Map exit General Call Information Area	page 271
ISZEREGRS	Register equates	page 272

ISZEENT macro **Description**

Used to establish the exit code CSECT name, AMODE(31), RMODE(ANY) and standard linkage with the caller (Session Manager) based on a supplied 72-byte save area (see also 'Address 8 – 72-byte register save area' on page 257). Establish a base register, and access to the exit work area.

Format

```

-----
name                               name: Symbol. Begin name in column 1
                                   Default: none
                                   One or more blanks must precede ISZEENT
ISZEENT
                                   One or more blanks must follow ISZEENT
-----
BASE=base register                 base register: register (2) - (12)
                                   Alternatively (R2) - (R12); R2 - R12; 2-12
                                   Default: R12
,EPLST=address list register      address list register: register (2) - (12)
                                   Alternatively (R2) - (R12); R2 - R12; 2-12
                                   Default: R10
                                   ,EWA=EWA register
                                   EWA register: register (2) - (12)
                                   Alternatively (R2) - (R12); R2 - R12; 2-12
                                   Default: R11

```

Parameters

name

Specifies the CSECT name for the exit code. No default, must be supplied.

BASE=base register

Specifies the base register for the exit code.

,EPLST=address list register

Specifies the register to be set to point to the list of addresses. A USING for ISZEPLST is generated.

,EWA=EWA register

Specifies the register to be used to address the exit work area.

ISZERET macro Description

Used to return control from the exit point to Session Manager.

Format

```

-----
ISZERET                                One or more blanks must precede ISZERET
ISZERET                                One or more blanks must follow ISZERET
-----
RC=rc register  rc register: register (0) - (12), (14) - (15)
                  Alternatively (R0) - (R12), (R14) - (R15)
RC=rc value     rc value: 0 or 40
                  Default: 0

```

Parameters

RC=*rc register*

Specifies a register containing an exit return code.

RC=*rc value*

Specifies a numeric value, either 0 or 40.

ISZESTOR macro Description

Used to request that Session Manager obtains or frees storage (see also 'Address 6 – Session Manager Storage Obtain/Free routine' on page 257 and 'Storage Obtain/Free routine' on page 258).

Format

```

-----
ISZESTOR                                One or more blanks must precede ISZESTOR
ISZESTOR                                One or more blanks must follow ISZESTOR
-----
TYPE=OBTAIN                             Default: none
TYPE=FREE
, EPLST=address of Exit Parameter List  'EPL address': register (0) - (15)
                                          Alternatively (R0) - (R15)
                                          'EPL address': symbol
, WORK=address of 3-word                'Work address': register (0) - (12)
      work area                        Alternatively (R0) - (R12)
                                          'Work address': symbol
                                          Default: see below
, A=area address                       area address: register (2) - (12)
                                          Alternatively (R2) - (R12)
                                          area address: symbol
, L=length                             length: numeric value
                                          length: register (2) - (12)
                                          Alternatively (R2) - (R12)
                                          length: symbol

```

Parameters

TYPE=OBTAIN

TYPE=FREE

Specifies whether storage is obtained or freed.

,EPLST=*address of Exit Parameter List*

Specifies the address of the Exit Parameter List (EPL). The EPL address can be held in a register or specified as a label (for example, ISZEPLST).

,WORK=*address of a 3-word work area*

Specifies the address of a 3-word area to be used to build the parameter list for the call to the Storage Obtain/Free routine. By default, a Session Manager area is reserved for this purpose at a fixed location. The work area address, if provided, can be held in a register or specified as a label.

,A=*area address*

Specifies, for TYPE=FREE, the address of the area to be freed. Session Manager does not require the length to be explicitly specified for TYPE=FREE, as the information is held within an area prefixing the area supplied upon successful completion of TYPE=OBTAIN. The area address can be held in a register or a fullword in storage.

,L=*length*

Specifies the length of storage to be OBTAINED. This value is rounded up to a 4-byte boundary, and will be limited to 16K. This parameter is not required for TYPE=FREE. A numeric value may be specified for the length, or it may be held in a register or a fullword in storage.

Output register information

Register	Contents
R0	Used as a work register
R1	The address of the allocated storage when TYPE=OBTAIN is successful, otherwise used as a work register
R2-R13	Unchanged
R14	Used as a work register
R15	Contains the return code

Return codes

When control is returned from Session Manager, Register 15 contains a return code. This is set as follows:

0 =	Request successful, R1 contains address of area obtained if TYPE=OBTAIN
4 =	Successful, but length truncated to 16K
8 =	Not successful
-4 =	Invalid request

ISZEVAR macro Description

Used to invoke the Session Manager Variable Access routine (see also ‘Address 2 – Session Manager Variable Access routine’ on page 255 and ‘Variable Access routine’ on page 259).

Format

ISZEVAR	One or more blanks must precede ISZEVAR
	One or more blanks must follow ISZEVAR
TYPE= <i>type</i>	<i>type</i> : GET or PUT Default: GET
,PARM= <i>area address</i>	<i>area address</i> : register (1) - (12) Alternatively (R1) - (R12) <i>area address</i> : symbol Default: (R1)
,EPLST= <i>address of Exit Parameter List</i>	‘ <i>EPL address</i> ’: register (0) - (15) Alternatively (R0) - (R15) ‘ <i>EPL address</i> ’: symbol
,WORK= <i>address of 3-word work area</i>	‘ <i>Work address</i> ’: register (0) - (12) Alternatively (R0) - (R12) ‘ <i>Work address</i> ’: symbol Default: see below

Parameters

TYPE=*type*

Specifies the type of access, GET or PUT.

,PARM=*area address*

Specifies the address of the Variable Access Parameter Block (see page 248). The area address can be held in a register or specified as a label.

,EPLST=*address of Exit Parameter List*

Specifies the address of the Exit Parameter List (EPL). The EPL address can be held in a register or specified as a label (for example, ISZEPLST).

,WORK=*address of a 3-word work area*

Specifies the address of a 3-word area to be used to build the parameter list for the call to the Variable Access routine. By default, a Session Manager area is reserved for this purpose at a fixed location. The work area address, if provided, can be held in a register or specified as a label.

Output information

See ‘Variable access’ on page 246 for details.

ISZECMD macro Description

Used to invoke the Session Manager command-issuing routine (see also ‘Address 7 – Session Manager command-issuing routine’ on page 257 and ‘Issuing Session Manager commands’ on page 260).

Format

ISZECMD	One or more blanks must precede ISZECMD
	One or more blanks must follow ISZECMD
,PARM= <i>area address</i>	<i>area address</i> : register (1) - (12) Alternatively (R1) - (R12) <i>area address</i> : symbol Default: (R1)
,EPLST= <i>address of Exit Parameter List</i>	‘ <i>EPL address</i> ’: register (0) - (15) Alternatively (R0) - (R15) ‘ <i>EPL address</i> ’: symbol
,WORK= <i>address of 2-word work area</i>	‘ <i>Work address</i> ’: register (0) - (12) Alternatively (R0) - (R12) ‘ <i>Work address</i> ’: symbol Default: see below

Parameters

PARM=*area address*

Specifies the address of the Command Area. See ‘Issuing Session Manager commands’ on page 260 for details on the format of the Command Area. The area address can be held in a register or specified as a label.

,EPLST=*address of Exit Parameter List*

Specifies the address of the Exit Parameter List (EPL). The EPL address can be held in a register or specified as a label (for example, ISZEPLST).

,WORK=*address of a 2-word work area*

Specifies the address of a 2-word area to be used to build the parameter list for the call to the command-issuing routine. By default, a Session Manager area is reserved for this purpose at a fixed location. The work area address, if provided, can be held in a register or specified as a label.

Output information

See ‘Issuing Session Manager commands’ on page 260 for details.

ISZEPLST macro Description

On entry to the exit point code, R1 points to an area containing the address of the Entry Parameter List (EPL). This macro, by default, generates a DSECT, ISZEPLST, which maps the Entry Parameter List (see also 'Entry Parameter List format' on page 254).

Format

```

-----
ISZEPLST          One or more blanks must precede ISZEPLST
ISZEPLST          One or more blanks must follow ISZEPLST
-----
P=prefix        prefix: Symbol. 1-3 character field name prefix
                  Default: EPL
, DSECT=YES      Default: DSECT=YES
, DSECT=NO

```

Parameters

P=*prefix*

Specifies a prefix value, from 1 to 3 characters, used to generate the name for each field in the DSECT. Must be coded such that a valid assembler symbol is generated (for example cannot start with a numeric).

, DSECT=YES
, DSECT=NO

Specifies whether a DSECT is generated, or space reserved for the fields.

Code generated by default

```

ISZEPLST DSECT ,
EPLGCI@ DS    A      Address of general call info area
EPLVARA@ DS    A      Address of Variable Access routine
EPLMSG@ DS    A      Address of output message area
EPLIPB@ DS    A      Address of exit input parameter blk
EPLRCAB@ DS    A      Address of return code/action blk
EPLSTOF@ DS    A      Address of Storage Obtain/Free routine
EPLCMDR@ DS    A      Address of command-issuing routine
EPLRSA@ DS    A      Address of 72-byte register save area
EPLRSVRD DS    16F    Reserved for future use

```

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

ISZEIPB macro Description

Address 4 of the parameter list addresses points to the Input Parameter Block. This macro, by default, generates a DSECT, ISZEIPB, which maps the Input Parameter Block (see also 'Address 4 – Input Parameter Block' on page 255).

Format

```

-----
ISZEIPB          One or more blanks must precede ISZEIPB
ISZEIPB          One or more blanks must follow  ISZEIPB
-----
P=prefix        prefix: Symbol. 1-3 character field name prefix
                  Default: IPB
,DSECT=YES      Default: DSECT=YES
,DSECT=NO

```

Parameters

P=prefix

Specifies a prefix value, from 1 to 3 characters, used to generate the name for each field in the DSECT. Must be coded such that a valid assembler symbol is generated (for example cannot start with a numeric).

,DSECT=YES
,DSECT=NO

Specifies whether a DSECT is generated, or space reserved for the fields.

Code generated by default

```

ISZEIPB DSECT ,
IPBEWA@ DS    A           Address of exit work area
IPBEWAL DS    F           Exit work area length
IPBIDST@ DS    A           Address of 3270 datastream
IPBIDSTL DS    F           Length of 3270 datastream
IPBFLAGC DS    X           Common flag
IPBSRUN EQU   X'80'       Session script is running
IPBMISR EQU   X'40'       MISER is active for the session
IPBFLAGI DS    X           Input exit flag
IPBAPPI EQU   X'80'       Input from application
IPBSMFI EQU   X'40'       Input from Session Manager function
IPBFLAGO DS    X           Output exit flag
IPBFLAGS DS    X           Spare flag
IPBUSRID DS    CL8        Userid
IPBTRMID DS    CL8        Terminal Id
IPBLOGM DS    CL8        Logmode name
IPBAPPLN DS    CL8        APPLID
IPBACBN DS    CL8        ACB name
IPBSESNO DS    F           Session number
IPBUTSKI DS    CL8        User task id
IPBSTSKI DS    CL8        Session task id
IPBTTSKT DS    CL4        Terminal task type
IPBBINDI DS    XL26       Bind image
IPBRSVRD DS    CL66       Reserved for future use

```

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

ISZERCAB macro Description

Address 5 of the parameter list addresses points to the output Return Code/Action Block. This macro, by default, generates a DSECT, ISZERCAB, which maps the output Return Code/Action Block (see also 'Address 5 – Return Code/Action Block' on page 256).

Format

```
-----
ISZERCAB          One or more blanks must precede ISZERCAB
                  One or more blanks must follow  ISZERCAB
-----
P=prefix          prefix: Symbol. 1-3 character field name prefix
                  Default: RAB
,DSECT=YES        Default: DSECT=YES
,DSECT=NO
```

Parameters

P=prefix

Specifies a prefix value, from 1 to 3 characters, used to generate the name for each field in the DSECT. Must be coded such that a valid assembler symbol is generated (for example cannot start with a numeric).

,DSECT=YES
,DSECT=NO

Specifies whether a DSECT is generated, or space reserved for the fields.

Code generated by default

```
ISZERCAB DSECT ,
RABRDST@ DS    A          Address of replacement 3270 datastream area
RABRDSTL DS    F          Length of replacement 3270 datastream area
          ORG    RABRDST@
RABAIDS@ DS    A          Address of application-input 3270 datastream area
RABAIDSL DS    F          Length of application-input 3270 datastream area
          ORG    ,
RABFLAGC DS    X          Common flag
RABSCHDS EQU   X'80'      Cause session script to be scheduled
RABHCOO EQU   X'40'      One-off hardcopy
RABFLAGI DS    X          Input exit flag
RABRIDST EQU   X'80'      Reject input datastream
RABUNLKK EQU   X'40'      Unlock keyboard
RABTRACC EQU   X'20'      Trace current datastream
RABUTRON EQU   X'10'      Turn on User datastream tracing
RABUTROF EQU   X'08'      Turn off User datastream tracing
RABSEND D EQU   X'04'      Send 3270 datastream to the terminal
RABSHCPI EQU   X'02'      Start hardcopying input screens
RABPHCPI EQU   X'01'      Stop hardcopying input screens
RABFLAGO DS    X          Output exit flag
RABRODST EQU   X'80'      Reject output datastream
```

RABAISS EQU	X'40'	Application-input datastream supplied
RABSHCPO EQU	X'02'	Start hardcopying output screens
RABPHCPO EQU	X'01'	Stop hardcopying output screens
RABFLAGS DS	X	Spare flag
RABSCRPT DS	CL8	Name of session script to schedule
RABSDAT@ DS	A	Address of datastream to be sent to the terminal
RABSDATL DS	F	Length of datastream to be sent to the terminal
RABHCPRF DS	CL8	HCPROFILE name
RABHCOPT DS	CL2	HCOPTION number
RABRSVRD DS	CL46	Reserved for future use

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

ISZEOMA macro Description

Address 3 of the parameter list addresses points to the Output Message Area. This macro, by default, generates a DSECT, ISZEOMA, which maps the Output Message Area (see also 'Address 3 – Output Message Area' on page 255).

Format

```
-----
ISZEOMA          One or more blanks must precede ISZEOMA
ISZEOMA          One or more blanks must follow  ISZEOMA
-----
P=prefix        prefix: Symbol. 1-3 character field name prefix
                  Default: OMA
,DSECT=YES      Default: DSECT=YES
,DSECT=NO
```

Parameters

P=prefix

Specifies a prefix value, from 1 to 3 characters, used to generate the name for each field in the DSECT. Must be coded such that a valid assembler symbol is generated (for example cannot start with a numeric).

,DSECT=YES
,DSECT=NO

Specifies whether a DSECT is generated, or space reserved for the fields.

Code generated by default

```
ISZEOMA DSECT ,
OMAMSGNO DS F          LOG message number (600-699)
OMAMSGDL DS F          LOG message length
OMAMTEXT DS CL132      LOG message text
OMASPARE DS CL4        Spare
OMARSVRD DS CL64      Reserved for future use
```

ISZEGCIA macro Description

Address 1 of the parameter list addresses points to the General Call Information Area. This macro, by default, generates a DSECT, ISZEGCIA, which maps the General Call Information Area (see also 'Address 1 – General call information area' on page 254).

Format

```

-----
ISZEGCIA          One or more blanks must precede ISZEGCIA
ISZEGCIA          One or more blanks must follow  ISZEGCIA
-----
P=prefix        prefix: Symbol. 1-3 character field name prefix
                  Default: GCI
, DSECT=YES      Default: DSECT=YES
, DSECT=NO

```

Parameters

P=*prefix*

Specifies a prefix value, from 1 to 3 characters, used to generate the name for each field in the DSECT. Must be coded such that a valid assembler symbol is generated (for example cannot start with a numeric).

, DSECT=YES
, DSECT=NO

Specifies whether a DSECT is generated, or space reserved for the fields.

Code generated by default

```

ISZEGCIA DSECT ,
GCICIND DS    CL3      Call indicator
GCISPR1 DS    CL1      Spare
GCIREASN DS    CL8      Call reason
GCIDATE DS    CL8      Date, dd/mm/yy or mm/dd/yy
GCIJDATE DS   XL4      Julian date, 00yydddF
GCITIME DS    CL8      Time, hh:mm:ss
GCITOD  DS   XL8      TOD clock value
GCISBLK@ DS    A       Address of statistics block
GCIRSVRD DS   CL64     Reserved for future use

```

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

ISZERECS macro Description

Generates register equates.

Format

```

-----
                                One or more blanks must precede ISZERECS
ISZERECS
                                One or more blanks must follow  ISZERECS
-----

```

Parameters

None.

Code generated

```

R0      EQU    0
R1      EQU    1
R2      EQU    2
R3      EQU    3
R4      EQU    4
R5      EQU    5
R6      EQU    6
R7      EQU    7
R8      EQU    8
R9      EQU    9
R10     EQU    10
R11     EQU    11
R12     EQU    12
R13     EQU    13
R14     EQU    14
R15     EQU    15

```

Input 3270 datastream exit point (E25)

Note The characteristics of the input/output exits differ from the other user exit points in several important respects. For details, see 'Input and output 3270 datastream exit points' on page 252.

The E25 exit point is invoked each time terminal input is received, whether for an application session, or for a Session Manager function (for example the Menu, DL0G, and so on). It may be used to modify the input datastream or supply an alternative datastream. Since the exit point is positioned prior to escape sequence processing, this exit point can cause or suppress escapes.

By setting the appropriate fields in the Return Code/Action Block (see page 256), this exit point can cause Session Manager to:

- Process an alternative, exit-supplied input datastream.
- Supply a 3270 datastream to be written to the terminal.
- Schedule the running of an exit-named session script (only when the input datastream is from an application).

- Reject the input datastream, and optionally send an 'unlock keyboard' datastream to the terminal.
- Hardcopy the input screen if the datastream is from an application.
- Start/stop hardcopying input/output datastreams for an application session.
- Perform datastream tracing (see 'Datastream tracing from the E25 exit' below).

Datastream tracing from the E25 exit

By setting the appropriate fields in the Return Code/Action Block (see page 256), the E25 exit can cause Session Manager to:

- Trace the current input datastream.
- Turn on datastream tracing for all input and output datastreams for the user.
- Turn off datastream tracing for the user.

Writing a message to the Audit log

By setting the appropriate fields in the Output Message Area (see page 255), this exit point can cause Session Manager to write an exit-supplied message to the Audit log.

Using Session Manager Windows and the E25 exit

When using Session Manager Windows to access an application, the E25 exit point will see the input datastreams for the Windows function; it will *not* see the input datastream passed by the Windows function to the application.

Depending on what the exit is designed for and, in particular, whether or not the design is sensitive to buffer addresses, the exit may require modification. Although the entered data will be the same as would be seen by the exit when the application is being accessed directly, the buffer addresses will be different. In fact, these buffer addresses can vary depending on where the application window has been positioned on the screen.

Session Manager routines available

These Session Manager routines are available at the E25 exit point:

- Variable Access routine – get, put, delete user-defined and certain Session Manager variables.
- Storage Obtain/Free routine.
- Routine to cause a Session Manager command to be issued (see 'Sample exit' on page 261).

Call information General Call Information (pointed to by address 1)

Call indicator 'E25'.

Reason code '3270IN'.

For other general call information, see page 254.

Variable Access routine (pointed to by address 2)

See page 255.

Output Message Area, may be modified by the exit (pointed to by address 3)

See page 255. Sends message to log.

Input Parameter Block (pointed to by address 4)

Position	Len	Description
1-4	4	Address of exit work area, or zero if EXITWALEN 0 specified. This area is initialized to X'00' on entry to the exit point.
5-8	4	Length of the exit work area, or 0.
9-12	4	Address of the input 3270 datastream.
13-16	4	Length of the input 3270 datastream.
17-17	1	Common flag byte: X'80': Session script is running. X'40': MISER is active for the application session (only set if byte 18 bit X'80' is on).
18-18	1	Flag byte: X'80': Terminal input for application session. X'40': Terminal input for Session Manager function.
19-19	1	Flag byte: Not used for E25 exit point.
20-20	1	Flag byte: Unused at present.
21-28	8	Userid. Always supplied.
29-36	8	Terminal Id. Always supplied.
37-44	8	VTAM Logmode name.
45-52	8	APPLid. Only supplied when terminal input for an application session as opposed to Session Manager function.
53-60	8	VTAM ACB name.
61-64	4	Session number, hexadecimal. Only supplied when terminal input for an application session as opposed to Session Manager function.
65-72	8	User task id, UAAAnnnn. Always supplied.
73-80	8	Session task id, SAAnnnnn. Only supplied when terminal input for an application session as opposed to Session Manager function.
81-84	4	Terminal task type.
85-110	26	VTAM bind image.
111-176	66	Reserved for future use.

Storage Obtain/Free routine (pointed to by address 6)

See page 257.

Command-issuing routine (pointed to by address 7)

See page 257.

Register save area (pointed to by address 8)

The address of a 72-byte area to be used as the exit's save area.

Return information**Return Code/Action Block (pointed to by address 5)**

Position	Len	Description
1-4	4	Address of the replacement input 3270 datastream.
5-8	4	Length of the replacement 3270 datastream.
9-9	1	Common flag byte: X'80': Cause session script to be run. If the input datastream is for an application session as opposed to a Session Manager function, and there is no currently active script on that session then the script identified in position 13-20 of the Return Code/Action Block is scheduled to run at the next opportunity.
10-10	1	Flag byte: X'80': Reject input datastream. X'40': When the input datastream is to be rejected, Session Manager is to send an 'unlock keyboard' datastream to the terminal. X'20': Session Manager is to trace the input datastream passed to (and possibly modified by) the input exit. X'10': Session Manager is to start tracing all input and output datastreams for the User, starting with the <i>next</i> datastream after the one which caused the input exit to run. X'08': Session Manager is to stop tracing all input and output datastreams for the User. X'04': Session Manager is to send a datastream to the terminal. The datastream is identified in positions 21-28 or the Return Code/Action Block. X'02': Hardcopy the current application session input screen. This will only be actioned if the datastream supplied to the E25 exit is from an application session. X'01': Start hardcopying application input screens for the session identified in the Input Parameter Block on entry to the E25 exit. This will only be actioned if the datastream supplied to the E25 exit is from an application session.
11-11	1	Flag byte: Not used for E25 exit with one exception – the flags used to start/stop hardcopying output screens may be set by the E25 exit, and will be actioned.
12-12	1	Flag byte: Unused at present.

Position	Len	Description
13-20	8	If the common flag byte, bit X'80', is set then the script identified in this field will be scheduled to run if no script is currently running on the session, and if a valid script name has been supplied.
21-24	4	Address of the datastream to be sent to the terminal when flag in position 10 is X'04'.
24-28	4	Length of the datastream to be sent to the terminal when flag in position 10 is X'04'.
29-36	8	HCPROFILE name. For a hardcopy request, override the user's hardcopy profile name.
37-38	2	HCOPTION number. For a hardcopy request, override the user's hardcopy option number.
39-84	46	Reserved for future use.

Output 3270 datastream exit point (E35)

Note The characteristics of the input/output exits differ from the other user exit points in several important respects. For details, see 'Input and output 3270 datastream exit points' on page 252.

The E35 exit point is invoked each time an output datastream is sent from an application session, but not for a Session Manager function (for example the Menu, DL0G, OLA, and so on). It may be used to modify the output datastream or supply an alternative datastream. Since the exit point is positioned prior to session script processing, this exit point may affect a session script's behavior (for example, by creating or modifying variables).

By setting the appropriate fields in the Return Code/Action Block (see page 256), this exit point can cause Session Manager to do the following:

- Process an alternative, exit-supplied output datastream.
- Schedule the running of an exit-named session script.
- Reject the output datastream, and optionally send an exit-supplied input datastream to the application.
- Hardcopy the application output screen.
- Start/stop hardcopying input/output datastreams for the application session.

Writing a message to the Audit log

By setting the appropriate fields in the Output Message Area (see page 255), this exit point can cause Session Manager to write an exit-supplied message to the Audit log.

Session Manager routines available

The following Session Manager routines are available at this exit point:

- Variable Access routine – get, put, delete user-defined and certain Session Manager variables.

- Storage Obtain/Free routine.
- Routine to cause a Session Manager command to be issued.

Call information **General Call Information (pointed to by address 1)**

Call indicator 'E35'.

Reason code '3270OUT'.

For other general call information, see page 254.

Variable Access routine (pointed to by address 2)

See page 255.

Output Message Area, may be modified by the exit (pointed to by address 3)

See page 255. Sends message to log.

Input Parameter Block (pointed to by address 4)

Position	Len	Description
1-4	4	Address of exit work area, or zero if EXITWALEN 0 specified. This area is initialized to X'00' on entry to the exit point.
5-8	4	Length of the exit work area, or 0.
9-12	4	Address of the output 3270 datastream.
13-16	4	Length of the output 3270 datastream.
17-17	1	Common flag byte: X'80': Session script is running. X'40': MISER is active for the application session.
18-18	1	Flag byte: Not used for E35 exit point.
19-19	1	Flag byte. Unused at present.
20-20	1	Flag byte. Unused at present.
21-28	8	Userid. Always supplied.
29-36	8	Terminal Id. Always supplied.
37-44	8	VTAM logmode name.
45-52	8	APPLid. Always supplied.
53-60	8	VTAM ACB name.
61-64	4	Session number, hexadecimal. Always supplied.
65-72	8	User task id, UAAnnnnn. Always supplied.
73-80	8	Session task id, SAAnnnnn. Always supplied.
81-84	4	Terminal task type.
85-110	26	VTAM bind image.

Position	Len	Description
111-176	66	Reserved for future use.

Storage Obtain/Free routine (pointed to by address 6)

See page 257.

Command-issuing routine (pointed to by address 7)

See page 257.

Register save area (pointed to by address 8)

The address of a 72-byte area to be used as the exit's save area.

Return Code/Action Block (pointed to by address 5)

Position	Len	Description
1-4	4	Address of the replacement output 3270 datastream, or Application-Input datastream (if the flag byte in position 10 of the Return Code/Action block is set to X'CO').
5-8	4	Length of the 3270 datastream.
9-9	1	Common flag byte. X'80': Cause session script to be run. If there is no currently active script on the session then the script identified in position 13-20 of the Return Code/Action block is scheduled to run at the next opportunity.
10-10	1	Flag byte: X'80': Reject output datastream. X'40': If output datastream is to be rejected, Session Manager is to send an 'Application-Input' 3270 datastream to the application. This datastream is identified by the fields in position 1-8 of the Return Code/Action block. X'02': Hardcopy the current application session output screen. X'01': Start hardcopying application output screens for the session identified in the Input Parameter Block on entry to the E35 exit.
11-11	1	Flag byte: Not used by the E35 exit point with one exception – the flags used to start/stop hardcopying input screens may be set by the E35 exit, and will be actioned.
12-12	1	Flag byte: Unused at present.
13-20	8	If the common flag byte is set (to X'80') then the script identified in this field will be scheduled to run if no script is currently running on the session, and if a valid script name has been supplied.
21-84	64	Reserved for future use.

Return information

The Multiple Exit Driver

In Session Manager, only one name needs to be specified for the user exit. This exit is invoked at a number of different points, as described in the previous section. Rather than code all the processing in one large exit, it may be more convenient to code several different exits. This then allows the processing for one exit point to be changed without affecting the others.

The Multiple Exit Driver provides the means to manage the loading and deleting of the chosen exits in the Session Manager system. A panel can be invoked which shows the status of each exit and update request commands can be entered to delete or reload any exit. If only a single exit point is used in the Session Manager system, the Multiple Exit Driver is not needed.

The Exit Control panel

Session Manager		Exit Control		dd/mm/yyyy hh:mm:ss		
LU	<i>tuname</i>	Module	Status	Date	Time	<i>user</i>
Exit						
No.	Description	Name	Status	Date	Time	Address
	Multiple Exit Driver	ISZEXT00	Active	04/12/02	17:04:11	80241560
EFH	S/Mgr File Handler	ISZEXTFH	New copy	04/12/02	17:04:11	802423E8
E01	S/Mgr initialization	ISZEXT01	Not found	04/12/02	17:04:11	
E05	Configuration	ISZEXT05	New copy	04/12/02	17:04:11	802432D8
E06	Application status	ISZEXT06	Not found	04/12/02	17:04:11	
E08	Timer driven	ISZEXT08	New copy	04/12/02	17:04:11	8023B628
E09	CALLEXIT routine	ISZEXT09	New copy	04/12/02	17:04:11	80244650
E11	VTAM logon	ISZEXT11	Not found	04/12/02	17:04:11	
E21	Signon validation	ISZEXT21	Active	04/12/02	17:04:11	80245280
E22	Signon completion	ISZEXT22	Not found	04/12/02	17:04:11	
E29	Signoff	ISZEXT29	Not found	04/12/02	17:04:11	
E31	Slave session start	ISZEXT31	Not found	04/12/02	17:04:11	
E33	Slave session started	ISZEXT33	New copy	04/12/02	17:04:11	80007020
E39	Slave session end	ISZEXT39	Not found	04/12/02	17:04:11	
E71	Update UserExit	ISZEXT71	Not found	04/12/02	17:04:11	
E79	Slave session switch	ISZEXT79	Active	04/12/02	17:04:11	8023B060
To load exit, put cursor in prefix area, hit PF10. To delete, hit PF11.						
====>						
PF1:Help PF3:Quit PF7:Bwd PF8:Fwd PF10:Reload PF11>Delete cursor selected exit						

This panel is invoked by entering the command DRIVER on the Menu panel, provided that the Menu panel has been set up correctly. The setup is described later in this chapter.

To load or reload an exit, place the cursor in the relevant command prefix area and press PF10. To remove an already active exit, place the cursor in the relevant command prefix area and press PF11.

Each status field can contain one of the following:

Active	The exit is in storage.
Deleted	The exit has been deleted from storage.
New Copy	A new copy of the exit has been loaded into storage.
Not Found	The exit was not found in the available libraries.
Abended	The Slave Exit has abended. A new copy will be loaded at the next call to the Multiple Exit Driver.

The update request commands are passed to the Multiple Exit Driver by a command script called DRIVEXIT and are actioned immediately.

Exits that have a status of Not Found or Deleted are not called by the Exit Driver, because a return code of 40 is issued. If an exit with such a status is loaded, the Exit Driver will return a code of 44 to Session Manager to allow all exit points to be called again; therefore reversing the effect of the previous return code 40.

Setup steps

The Menu panel for users responsible for controlling the loading and deleting of exits must be changed to allow invocation of the Control panel. In the PROCESS section of the Menu panel, include:

```

If t_command = 'DRIVER' then      /* Command keyed on Menu
    Let uc_oldpan = T_panel        /* Save original panel name
    Let t_panel = 'DRIVER'        /* Set Control panel name
    Let t_command = ' '          /* Clear command
End

```

The command could be something different, it does not have to be DRIVER. By carefully choosing which Menus to change, access to the Control panel can be restricted to only those users entitled to delete or load exits. This is important because *once a request has been issued from the Control panel, the request is processed at the next exit invocation whether or not the user has update authority.*

The name of the Exit Driver is ISZEXT00. This is therefore the name that should be specified on the EXIT parameter of the OPTION configuration statement:

```
OPTION EXIT ISZEXT00
```

The module names of all exits that are to be used with the Exit Driver must conform to a standard. This is:

```
ISZEXTnn
```

where nn is the exit point number. For example, ISZEXT05 would be the configuration statement processing exit. The supplied sample exits already have the correct names. Any Installation-specified exit can be used as it is, other than to give it the correct name. If an existing exit uses several of the exit points, a copy can be made for each exit point. However, it may be preferable to divide this exit into several different exits.

The Exit Driver and the ISZEXTnn exit modules must reside in a library that is specified in the startup for the Session Manager system. This applies to new copies of any exit as well as to the original versions.

How the Multiple Exit Driver works

At Session Manager initialization, the Exit Driver attempts to load all the exits specified in its internal table, EXTTAB. The internal table should never need to be changed.

When the system is running, the Exit Driver is invoked with call indicators relating to the exit points as described in the previous chapter. The Exit Driver checks that the appropriate exit is loaded and active. If it is, it calls the exit with a dummy parameter call area which contains the original calling parameters. If the exit is not loaded, control is returned to Session Manager. The Exit Driver uses return codes decimal 40 and 44 to control what is being called. (40 requests no further calls and 44 requests calls for all call indicators).

If a new copy of the Exit Driver is requested by issuing the command 'UPDATE EXIT ISZEXT00', all the active exits are called first with an indicator of E71 (update user exit point). This enables the exits to terminate normally before they are deleted from storage. When the new copy of the Exit Driver has been loaded, the driver attempts to load an exit for each exit point, in the same way as it does at startup.

At system closedown, the exits are called with an indicator of E99 (Session Manager termination exit point). This enables the exits to tidy up and allows storage to be freed.

Exit scripts

An Exit script may be defined to run in place of, or in addition to, most of the Assembler/COBOL exit points of the Session Manager user exit.

For most exit scripts, this is achieved by specifying the exit point name and the option 'S' on the OPTION configuration statement. A script of the form EXITxx must be defined on the SCRIPT statement for the relevant exit point, where xx is the exit point. For example, if E33 S is specified on the OPTION statement, a script EXIT33 will be called. A script can also cause the exit to run after the script has completed.

Exit parameters

An Exit script can access the exit parameters via the following exit parameter variables.

Notes

- 1 To access the date and time, use the t_date (or t_date_l) and t_time variables.
- 2 The second user exit parameter, the Variable Access routine, is not required in Exit scripts.
- 3 The third user exit parameter, the message parameter, is not required in Exit scripts. To send a message the following is required:
 - To send a message to the Audit log, use the AUDITMSG SCRIPT parameter.
 - To send a message to a user's screen, assign the required message to the t_message variable using a LET TPSL verb. The message is displayed as message number 226.

Valid exit SCRIPT parameters

Exit scripts can use the following SCRIPT parameters:

- All TPSL verbs
- AUDITMSG
- CALLEXIT
- ISZCMD
- CALL
- OUTSCAN (E26, E36 exit scripts only)

Use of ISZCMD

The Exit script is invoked with a user authority of 9. This provides access to the full set of available commands, even if the exit is invoked for a user who does not have the required authority.

The Exit script can invoke the following commands using ISZCMD:

BLOCK, BRECEIVE, BROADCAST, CLOSEDOWN, DELETE, DISCONNECT, DUMP, END, FLASH, FORCE, HALTSCRIPT, HARDCOPY, HCOPTION, LOCK, LOGOFF, MSG, PASSFREE, PCTransFER, QQUIT, RECORD, RESET, SPIN, STARTLINK, STOP, STOPACB, STOPLINK, TERMINATE, TRACE, TTPSL, UPDATE.

Return codes

The script *must* set the variable `t_rc` to the appropriate exit return code. Valid return codes are the same as those for the Assembler/COBOL user exit (see page 205).

Note If you want to call the user exit after an Exit script completes, set `t_rc` to 52. The user exit is called with the same parameters as were passed to the script.

User audit facility

Introduction

This facility provides an audit trail of Session Manager users activity for specific applications. This will be provided in the form of SMF records and will log the following:

- When the session is started by a user.
- Any input keyed in by the user during the session
- When the user ends the session.

Example exits are shipped with the product, as specified below. These are driven for the above events. The SMF record type and VTAM applid name(s) are hard coded in the exits but the user can modify these to meet their own requirements.

E33 Session post-initiation exit.

This is invoked when the session is established and will write away an SMF record if the session is for one of the specified VTAM applid's. The record contains the following data:

Field Ref	Length	Comments
SMFRLEN	2	Record length 40
SMFRSEG	2	Segment descriptor (not set)
SMFRFLAG	1	Flags (reserved)
SMFRTYPE	1	SMF record type. Values in the range 128-255 can be used but code will default to 240
SMFRTIME	4	Time since midnight (100ths of a second)
SMFRDATE	4	Date X'0cyydddF'
SMFRSID	4	System identifier
SMFRSTYP	1	Record sub-type. X'01' (session start)
	1	Reserved
SMFRUSER	8	Userid
SMFRTERM	8	Terminal name - LU name
SMFRAPPL	8	VTAM applid

E39 Session termination exit

This is invoked when the session is ended and will write away an SMF record if the session is for one of the specified VTAM applid's. The record contains the following data:

Field Ref	Length	Comments
SMFRLEN	2	Record length 40
SMFRSEG	2	Segment descriptor (not set)
SMFRFLAG	1	Flags (reserved)
SMFRTYPE	1	SMF record type. Values in the range 128-255 can be used but code will default to 240
SMFRTIME	4	Time since midnight (100ths of a second)
SMFRDATE	4	Date X'0cyyddF'
SMFRSID	4	System identifier
SMFRSTYP	1	Record sub-type. X'03' (session end)
	1	Reserved
SMFRUSER	8	Userid
SMFRTERM	8	Terminal name - LU name
SMFRAPPL	8	VTAM applid

E25 Input 3270 datastream exit

This is invoked when terminal input is received for the session and will write away an SMF record if the session is for one of the specified VTAM applid's. The record contains the following data:

Field Ref	Length	Comments
SMFRLEN	2	Record length. Variable.
SMFRSEG	2	Segment descriptor (not set)
SMFRFLAG	1	Flags (reserved)
SMFRTYPE	1	SMF record type. Values in the range 128-255 can be used but code will default to 240
SMFRTIME	4	Time since midnight (100ths of a second)
SMFRDATE	4	Date X'0cyyddF'
SMFRSID	4	System identifier
SMFRSTYP	1	Record sub-type. X'02' (terminal input)
	1	Reserved
SMFRUSER	8	Userid
SMFRTERM	8	Terminal name - LU name
SMFRAPPL	8	VTAM applid
SMFRTIPL	2	Length of terminal input. Variable.
SMFRTIPT	nnnn	Terminal input.

Customizing and installing the exits

The shipped configuration will include new components (all in .SISZCONF) to provide this function.

- ISZE25AT (source code for the E25 terminal input exit)
- ISZE33AT (source code for the E33 session start exit)
- ISZE39AT (source code for the E39 session end exit)
- ISZATTAB (source code for a macro used by the exits)

Customization

Customization is required to assign a value for the SMF record type and also to specify which application(s) are to be audited.

This is accomplished by editing the supplied ISZATTAB macro:

```

MACRO
  ISZATTAB
ATSMFTYP EQU 240                SMF record type.
ATNUMENT DC  A(ATAPLNUM/ATENTLEN)  Number of following entries.
ATAPPLS  DS  OF                  VTAM APPLIDs to audit.
          DC  CL8'VTAMAP01'        Audit on this APPLID.
ATENTLEN EQU *-ATAPPLS          Length of entry.
          DC  CL8'VTAMAP02'        Audit on following APPLIDs.
          DC  CL8'VTAMAP03'
ATAPLNUM EQU *-ATAPPLS          Length of table.
MEND

```

The SMF record type is defined by label ATSMFTYP and the applications to be audited are specified by defining their VTAM applid in the table beginning at label ATAPPLS.

So if you want to assign a value of 241 for the SMF record type and to audit four applications which have the VTAM applid's of CICSAP01, CICSAP02, CICSAP03 and CICSAP04 you would amend ISZATTAB as follows:

```

ATSMFTYP EQU 241                SMF record type.
ATNUMENT DC  A(ATAPLNUM/ATENTLEN)  Number of following entries.
ATAPPLS  DS  OF                  VTAM APPLIDs to audit.
          DC  CL8'CICSAP01'        Audit on this APPLID.
ATENTLEN EQU *-ATAPPLS          Length of entry.
          DC  CL8'CICSAP02'        Audit on following APPLIDs.
          DC  CL8'CICSAP03'
          DC  CL8'CICSAP04'
ATAPLNUM EQU *-ATAPPLS          Length of table.

```

Installing the exits

Exits are supplied already assembled and linked, but if they need modifying then this is accomplished by the assemble and link edit of the provided source members (see sample JCL ISZEASM in library .SISZCONF). These must be linked into an APF authorized library which needs to be specified in the startup STEPLIB concatenation for the Session Manager system.

- Assemble and link ISZE00DR allocating the load module the name ISZEXT00. This is the multiple exit driver and is required to run the E33 and E39 exits.

- Assemble and link ISZE25AT allocating the load module the name ISZEXT25. ISZE25AT is not controlled by the ISZE00DR exit driver but needs ISZEXT25 to be specified on the INPUTEXIT parameter of the SYSTEM statement (see 'Input and output 3270 datastream exit points' on page 252).
- Assemble and link ISZE33AT allocating the load module the name ISZEXT33.
- Assemble and link ISZE39AT allocating the load module the name ISZEXT39.

Timeout exits facility

Introduction

This facility allows an exit script to be run when either of the IDLELOGOFF (E26), IDLEDISC (E26), IDLELOCK(E26) or SIDLTIME (E36) timers are about to expire. Action that the exit point can take is to either allow the timeout to proceed, or to cause the timeout period to be extended for a further timeout period.

Sample exits are shipped with the product as specified below.

Customizing and installing the exits

The shipped configuration will include new components (all in .SISZCONF).

- ISZE26IT (source code for the E26 timeout exit)
- ISZE36IS (source code for the E36 timeout exit)

Customization

The sample exits show how to extend or honour the timeout period, depending on the Userid, or certain contents of the visible application screen. Amend the exit script source to installation requirements.

Installing the exits

To include both exits, code the OPTION configuration statement as follows:

```
OPTION E26 S E36 S
```

To dynamically add the exits, copy the customized exit code to your Session Manager configuration with a member name of EXITnn, and issue UPDate Member EXITnn command. Then issue UPDate Enn S command, where nn is 26 or 36.

CHAPTER 9

Defining security and implementing dynamic menus

Security can be utilized to authenticate a user and to determine a user's IBM Session Manager for z/OS capabilities.

The subjects covered in this chapter are:

- 'Defining security' on page 290
- 'Implementing Dynamic Menus' on page 306

Defining security

Designing the signon panel

The signon panel provides an entry point to the Session Manager system and allows for various measures of security to be implemented. The security may take the form of passwords supplied by USER statements, from a VSAM file, or from an ESM which is accessed using either the ISZE21SF exit provided with Session Manager (source in library .SISZCONF, executable in library .SISZAUTH) or the ISZE21PH exit script provided with Session Manager (source in library .SISZCONF).

The signon panel used for the system is defined using the SIGNONPANEL parameter of the SYSTEM statement. If this parameter is not specified, a default panel called 'SIGNON' is required.

Three sample signon panels can be found in member ISZLENPU, which may be referenced using COPY ISZLEN for Classic and ISZLEN2 for OLA. These are SIGNON (an example signon panel), SIGNONE (an example signon panel that can be presented twice to allow for verification of new passwords), and SIGNONP (an example signon panel that allows password phrases to be entered). When using the ISZE21PH exit script and password phrases, the signon panel should be designed using the SIGNONP sample panel.

The sample signon panels can be amended to include your company logo, or several different signon screens can be created to cater for different needs. The default panel can be referenced by using the SIGNONPANEL parameter on the SYSTEM statement, but terminal-specific and profile-specific panels can also be specified on the PROFILE and TERMINAL statements.

Full descriptions of the SYSTEM statement and SIGNONPANEL parameter can be found in the *Technical Reference*; details of the PANEL statement and related parameters can be found in the *Panel, Script and Variables* manual.

Sample panel layouts – illustrating the code that can be used to generate different signon panels, and explaining the Panel and Script Language (TPSL) – can be found in the *Panels, Scripts and Variables* manual.

These, along with the samples provided in source form on the distribution media, may be used as a basis for creating Installation-specific panel designs.

User authentication

You should choose between:

- 1 using passwords specified on the USER statement. This is the least secure method of controlling access to Session Manager.
- 2 validating userids and passwords using VSAM files. This is the best method to use if you do not have a dedicated External Security Manager (ESM).
- 3 using one or more ESMs, such as RACF, to authenticate users and to determine a user's Session Manager capabilities. This is probably the best method if your users are already set up in an ESM.

Internal user authentication – the USER statement

The USER statement is one of the configuration statements that affect how Session Manager runs, and is used to control settings that apply to one particular user. One of the parameters to the USER statement is PASSWORD. This parameter sets the password that the user must enter in order to access Session Manager.

The PASSWORD parameter can be set either in the configuration file directly or, if used, through OLA. The user's Session Manager capabilities will also be determined by their Session Manager configuration settings.

External user authentication – VSAM and ESM

Security procedures for users signing on to Session Manager can be externalized through the use of the Session Manager user exit. The Signon Validation Exit point (E21) enables an Installation-generated security routine to be used. This method causes the normal Session Manager configuration file userid and password checking either to be bypassed completely, or to be used in conjunction with the additional external routine.

For details of all the exits supplied on the Session Manager distribution media, see the ISZCEXIT member on the distribution media.

Two E21 exits (ISZE21SF – source in library .SISZCONF, executable in library .SISZAUTH; and the new ISZE21PH exit – source in library .SISZCONF) are supplied with the product that utilizes the ESM and each can be used as supplied to authenticate users. ISZE21SF may also be modified to meet individual customer requirements. The supplied exits can also be used in conjunction with settings on the SECURITY parameter on the SYSTEM statement (see *Technical Reference* manual) to additionally determine a user's Session Manager capabilities.

The Signon Validation Exit point enables an Installation to make use of its own internally generated security routine. To use this facility, the EXIT parameter must be specified on the OPTION statement, and a suitably coded exit module must be in a library accessible to Session Manager. The exit module is usually loaded at initiation time and remains resident throughout the Session Manager session, unless reloaded or removed by the Update facility.

The exit point is called each time a user attempts to sign on to the system. The signon userid and password are passed to the exit, with many additional fields. It is the responsibility of the User Exit to perform the security clearance checking for each user, and therefore, it may be used to prevent unauthorized personnel from obtaining access to the Session Manager system.

For more detailed information on Session Manager Exits, please refer to 'Session Manager user exit processing' on page 197.

Sample user exits

Installations requiring their own exit routine should see ‘Session Manager user exit processing’ on page 197 which contains a full description of how such an Exit Routine should be used.

Sample User Exits are supplied on the distribution media. These are:

ISZE21VM	VSAM security exit
ISZE22VM	VSAM security exit
ISZE22DM	External security manager (ESM) for Dynamic Menus
ISZE21SF	External Security Manager (ESM) exit for User Verification

They are all supplied in source form, and should be assembled and linked in the normal manner and some are supplied assembled and linked in an executable format. The module name should be specified on the EXIT parameter of the OPTION configuration control statement. However, if more than one exit is to be used concurrently, the Multiple Exit Driver can be used to manage the exits. This means that the exit module names must conform to the standard expected by the Exit Driver, that is, ISZEXTnn where nn is the exit point number. See ‘The Multiple Exit Driver’ on page 279 for more details.

Exit script ISZE21PH

For Session Manager v3.1.00 and higher, a new sample E21 exit script is supplied. Script ISZE21PH provides all the functionality of the current E21 assembler exit and additionally supports:

- External Security Manager password phrases.
- Multi threaded ISM signon.
- The ability to override any multiple profiles assigned during user configuration, up to a maximum of eighteen.

ISZE21PH must be installed in place of any E21 assembler exit as the two are mutually exclusive. It is recommended that the exit script is used and not the assembler exit as any future enhancements will be provided in the exit script only.

To install the E21 exit script for OLA, either copy source member ISZE21PH from the supplied .SISZCONF library into the customer ASCRIP library, .SISZCASC, or copy source member ISZE21PH from the supplied .SISZCONF library into the customer library .SISZCCNF, then insert a COPY statement for ISZE21PH in each ISZCONxx member, and add the COPY ISZE21PH after the COPY ISZCOMON at the end of the ISZCONxx member in the .SISZCCNF library. (See also the *Online and Batch Administration* manual.)

To install the E21 exit script for Classic, copy source member ISZE21PH from the supplied .SISZCONF library into the customer library .SISZCUST. Then insert a COPY statement for ISZE21PH in each ISZCONxx member, and add the COPY ISZE21PH at the end of the ISZCONxx member in the .SISZCUST library. When using the ISZE21PH exit script and password phrases the configuration parameter SIGNONPANEL must also be modified to assign a signon panel that has been based on the supplied sample panel SIGNONP.

Using VSAM

The ISZE21VM and ISZE22VM sample user exits will be the basis of your security configuration if you decide to use VSAM to control user authorization.

ISZE21VM and ISZE22VM exits

ISZE21VM and ISZE22VM process each signon start and signon completion (exits points E21 and E22). They provide a VSAM-based security system which can be used when a proprietary security package is not available.

The exits validate each userid and password, or new password, against a VSAM KSDS and allocate a profile name. ISZE21VM reads a user record from the VSAM file and updates the profile for the user with application names extracted from the VSAM record. This gives a dynamic profile. This exit also contains an administrator function to maintain the VSAM file online. For more information, see 'Using the Session Manager ADMIN panel' below.

Using the Session Manager ADMIN panel

The ADMIN panel, contained within ISZC21VM (in library .SISZCONF), is used to maintain the userids and related information.

The following commands are available from the ADMIN panel:

GET	Get user information from the VSAM file and display the details on the panel.
UPD	Rewrite a record with the updated user information.
ADD	Create a new user record from the information entered on the panel.
DELETE	Remove the user from the VSAM file.

Using an External Security Manager (ESM)

If you decide to use an ESM, such as RACF, the ISZE21SF or ISZE21PH exit should be used. These exits use the z/OS System Authorization Facility (SAF) to obtain information from whichever ESM is available.

Two E21 exits (ISZE21SF exit – source in library .SISZCONF, executable in library .SISZAUTH; and the new ISZE21PH exit – source in library .SISZCONF) are supplied with the product that utilizes the ESM and each can be used as supplied to authenticate users or they may be modified to meet individual customer requirements. The supplied exits can also be used in conjunction with settings on the SECURITY parameter on the SYSTEM statement (see *Technical Reference* manual) to additionally determine a user's Session Manager capabilities.

Sample exits ISZE21SF and ISZE21PH are mutually exclusive and cannot be run collectively. Both exits support:

- Userid/password verification.
- Password update at Session Manager signon.
- Configurable option to assign the user's OLACCLASS, based on rules held in the ESM.

- Configurable option to assign the user's AUTHORITY, based on rules held in the ESM.
- Configurable option to check with the ESM that the user has READ access to the Session Manager ACB name.
- Configurable option to check with the ESM that the user has READ access to the terminal being used to signon from.

The ISZE21PH exit also supports:

- Userid/passphrase verification.
- Passphrase update at Session Manager signon.
- Enables the signon process to be multi-threaded.
- The ability to assign multiple profiles to a user, up to a maximum of eighteen.

For further explanation and examples of how to set up these security features see 'FAQs' below.

Also by installing and configuring the supplied E22 exit (ISZE22DM – source in library .SISZCONF, executable in library .SISZAUTH) the ESM can also be used to provide users with Dynamic Menus, containing only the sessions they are entitled to use – see 'Implementing Dynamic Menus' on page 306.

The ESM classes and resource names to be interrogated by the supplied E21 and E22 exits are configured by using the SECURITY parameter on the SYSTEM statement (see *Technical Reference* manual).

FAQs

How do I authenticate a user and their password?

To enable an ESM to be used to authenticate a user and their password appropriate resources must be defined in the ESM. The example below illustrates how to define the users and their passwords and how to define RACF groups and allocate the users to their corresponding RACF group in a RACF environment using the supplied members ISZRDEF2 and ISZRDEF3 (in library .SISZCONF) and how to configure the required E21 exit.

Example of defining users and their passwords to RACF

This example will add users AAAAA, BBBB, CCCCC, DDDDD, EEEEE, FFFFF, GGGGG, HHHHH and IIII to your Session Manager configuration and to IBM's RACF configuration. It will also create and configure the Session Manager exit points and define the Session Manager exit configuration.

The following users will be defined within RACF and Session Manager:

AAAAA	Password AAAAA	RACF group ISZAAAAA
BBBBB	Password BBBBB	RACF group ISZBBBBB
CCCCC	Password CCCCC	RACF group ISZCCCCC
DDDDD	Password DDDDD	RACF group ISZDDDDD
EEEEE	Password EEEEE	RACF group ISZEEEE
FFFFF	Password FFFFF	RACF group ISZFFFFF

GGGGG	Password GGGGG	RACF group ISZGGGGG
HHHHH	Password HHHHH	RACF group ISZHHHHH
IIIII	Password IIIII	RACF group ISZIIIII

Although the following steps will work as supplied most sites will have to modify them to meet their own site standards, especially their RACF security standards.

- 1 Define the RACF users and groups and allocate each user a default RACF Group.

Tailor to your installation standards and submit the supplied JCL member ISZRDEF2 (in library .SISZCONF). The following users will be defined to RACF with their corresponding default RACF group:

AAAAA	Password AAAAA	RACF group ISZAAAAA
BBBBB	Password BBBBB	RACF group ISZBBBBB
CCCCC	Password CCCCC	RACF group ISZCCCCC
DDDDD	Password DDDDD	RACF group ISZDDDDD
EEEEE	Password EEEEE	RACF group ISZEEEE
FFFFF	Password FFFFF	RACF group ISZFFFFF
GGGGG	Password GGGGG	RACF group ISZGGGGG
HHHHH	Password HHHHH	RACF group ISZHHHHH
IIIII	Password IIIII	RACF group ISZIIIII

- 2 If using the ISZE21SF or the ISZE22DM exit, either ensure the supplied ISZE00DR is in your Session Manager library, .SISZAUTH, or, if required, modify and assemble the supplied E00 Driver exit ISZE00DR.

Using the supplied JCL member ISZEASM follow the instructions at the top of the member and change to assemble and link the E00 Driver exit. Tailor to your installation standards and submit the JCL. Ensure that the exit is assembled and linked into a library that is included in your Session Manager JCL.

- 3 If using the ISZE21SF, ensure that the supplied ISZE21SF exit is in your Session Manager library, .SISZAUTH, or, if required, modify and assemble the supplied E21 user exit ISZE21SF.

Using the supplied JCL member ISZEASM follow the instructions at the top of the member and change to assemble and link the E21 SAF Signon exit. Tailor to your installation standards and submit the JCL. Ensure that the exit is assembled and linked into a library that is included in your Session Manager JCL.

If using the ISZE21PH exit script, for OLA either copy source member ISZE21PH from the supplied .SISZCONF library into the customer ASCRIP library, .SISZCASC, or copy source member ISZE21PH from the supplied .SISZCONF library into the customer library .SISZCCNF, then insert a COPY statement for ISZE21PH in each ISZCONxx member, add the COPY ISZE21PH after the COPY ISZCOMON at the end of the ISZCONxx member in the .SISZCCNF library. (See also the *Online and Batch Administration* manual.) To install the E21 exit script for Classic, copy source member ISZE21PH from the supplied .SISZCONF library into the customer library .SISZCUST. Then insert a COPY statement for ISZE21PH in each ISZCONxx member, add the COPY

ISZE21PH at the end of the ISZCONxx member in the .SISZCUST library. When using the ISZE21PH exit script and password phrases the configuration parameter SIGNONPANEL must also be modified to assign a signon panel that has been based on the supplied sample panel SIGNONP.

- 4 If using the ISZE21SF exit, update the OPTION statement with the E00 Driver exit point.

```
OPTION EXIT ISZEXT00
```

If using the ISZE21PH exit script, update the OPTION statement with the E21 exit script.

```
OPTION E21 S
```

- 5 Add users AAAAA to IIIII to your Session Manager configuration.

If not already incorporated then ensure that the supplied member ISZCRUSR (in library .SISZCONF) is incorporated into your Session Manager system.

- 6 For Classic systems:

Add COPY ISZCRUSR to your Session Manager configuration.

- 7 For OLA systems:

Tailor to your installation standards and submit the supplied JCL member ISZRDEF3 (in library .SISZCONF). This job will use the ISZLOJOB facility to add the users to your OLA USER dataset.

Note The supplied JCL member ISZRDEL2 (in library .SISZCONF) can be used to delete the users and their groups.

How do I set a user's Profiles?

To enable an External Security Manager (ESM) to be used to control users' Profiles, appropriate resources must be defined in the ESM and users given access to these resources. The example below illustrates how to define the ESM class, resources and users' access levels in a RACF environment using the supplied members ISZRDF13 and ISZRDF14 (in library .SISZCONF) and how to configure the E21 exit. The E21 exit is not required when using the ESM to determine users' Profiles but it is assumed in these examples that it would be used for user authentication.

The class and resource names to be used when querying the ESM are configured via the ESMPRFCLNM, the ESMPRFRSNM and the ESMPRFACC sub-parameters on the SECURITY parameter on the SYSTEM statement (see *Technical Reference* manual).

The setting of the ESMPRFCLNM sub-parameter is used to indicate that Session Manager should use the ESM to determine users' Profiles.

Note Session Manager ignores the PROF setting on USER statement if this sub-parameter is set.

Example of setting a user's Profiles

The example below assumes that 'Example of defining users and their passwords to RACF' (see page 294), defining the users and exit points, has already been applied and that the users, RACF groups and E21 exit have already been defined within RACF and Session Manager.

This example will define the RACF class to be used and define the RACF resources and their group permissions. It will also define the Session Manager Security settings.

After signing on to Session Manager the users will have the following Profiles assigned:

AAAAA	Profile ISZPAAAA
BBBBB	Profile ISZPBBBB
CCCCC	Profile ISZPCCCC
DDDDD	Profile ISZPDDDD
EEEEE	Profile ISZPEEEE
FFFFF	Profile ISZPFFFF
GGGGG	Profile ISZPGGGG
HHHHH	Profile ISZPHHHH
IIIII	Profile ISZPIIII

Although the following steps will work as supplied, most sites will have to modify them to meet their own site standards, especially their RACF security standards:

- 1 Add the supplied PROFILE and APPLs to your Session Manager configuration.

If not already incorporated then ensure that the supplied members ISZCRPRF and ISZCRAPL (in library .SISZCONF) are incorporated into your Session Manager system.

For Classic systems:

Add COPY ISZCRPRF and COPY ISZCRAPL to your Session Manager configuration.

For OLA systems:

Tailor to your installation standards and submit the supplied JCL member ISZRDF10 (in library .SISZCONF). This job will use the ISZLOJOB facility to add the PROFILEs and APPLs to your OLA PROFILE and APPL datasets.

- 2 Define the ISZPROF class to the dynamic CDT in RACF.

Tailor to your installation standards and submit the supplied JCL member ISZRDF13 (in library .SISZCONF). The ISZPROF will be defined to RACF. The POSIT value must be unique – please consult your RACF systems programmer. If your installation has already defined a dynamic CDT then remove the SETROPTS CLASSACT(CDT) RACLIST(CDT) command.

Alternatively an existing class like FACILITY could be used instead. If the FACILITY class is used then all the examples will need updating accordingly.

Note The supplied JCL member ISZRDL13 (in library .SISZCONF) can be used to delete the ISZPROF class from the dynamic CDT.

- 3 Define the resources to the RACF class ISZPROF and permissions to the RACF groups.

Tailor to your installation standards and submit the supplied JCL member ISZRDF14 (in library .SISZCONF). The following resources will be defined in the RACF class ISZPROF and the corresponding RACF groups defined and given READ access:

ISZ.PROFILE.ISZPAAAA	ISZAAAAA group
ISZ.PROFILE.ISZPBBBB	ISZBBBBB group
ISZ.PROFILE.ISZPCCCC	ISZCCCCC group
ISZ.PROFILE.ISZPDDDD	ISZDDDDD group
ISZ.PROFILE.ISZPEEEE	ISZEIEEE group
ISZ.PROFILE.ISZPFFFF	ISZFFFFFF group
ISZ.PROFILE.ISZPGGGG	ISZGGGGG group
ISZ.PROFILE.ISZPHHHH	ISZHHHHH group
ISZ.PROFILE.ISZPIIII	ISZIIIII group

- 4 Update the SECURITY sub-parameters on the SYSTEM statement as follows:

```
ESMPRFCLNM = ISZPROF
ESMPRFRSNM = ISZ.PROFILE.
```

Note The trailing dot must be present on the ESMPRFRSNM settings.

See *Technical Reference* manual for more details of these SECURITY sub-parameters.

What happens if the ESM prevents access to all profiles for my userid ?

Session Manager will assign the default profile as specified under the SYSTEM DEFPROFILE parameter.

How do I set a user's AUTH level and OLAClass?

To enable an ESM to be used to control Session Manager authorization levels and OLA classes, appropriate resources must be defined in the ESM and users given access to these resources. The example below illustrates how to define the ESM class, resources and user's access levels in a RACF environment using the supplied members ISZRDEF4 and ISZRDEF5 (in library .SISZCONF) and how to configure the required E21 exit.

The class and resource names to be used when querying the ESM are configured by the AUTHCLASSNAME, the AUTHRESNAME and the OLARESNAME sub-parameters on the SECURITY parameter on the SYSTEM statement (see Technical Reference manual).

It is by configuring the AUTHCLASSNAME that indicates to the supplied E21exit that Session Manager should check for ACB authorization within the ESM.

Example of setting a user's AUTH level and OLAClass.

The example below assumes that 'Example of defining users and their passwords to RACF' (see page 294), defining the users and exit points, has already been applied and that the users, RACF groups and E21 exit have already been defined within RACF and Session Manager

This example will define the RACF class to be used and define the RACF resources and their group permissions. It will also define the Session Manager Security settings.

After signing on to Session Manager the users will have the following AUTH and OLAClass assigned

AAAAA	Auth 9	OLA class IM
BBBBB	Auth 8	OLA class IM
CCCCC	Auth 7	OLA class AD
DDDDD	Auth 6	OLA class LA
EEEEE	Auth 5	OLA class LA
FFFFF	Auth 4	OLA class SU
GGGGG	Auth 3	OLA class US
HHHHH	Auth 2	OLA class US
IIIII	Auth 1	OLA class NO

Although the following steps will work as supplied most sites will have to modify them to meet their own site standards, especially their RACF security standards.

- 1 Define the ISZCLASS class to the dynamic CDT in RACF.

Tailor to your installation standards and submit the supplied JCL member ISZRDEF4 (in library .SISZCONF). The ISZCLASS will be defined to RACF. The POSIT value must be unique - please consult with your RACF systems programmer. If your installation has already defined a dynamic CDT then remove the SETROPTS CLASSACT(CDT) RACLIST(CDT) command.

Alternatively an existing class like FACILITY could be used instead. If the FACILITY class is used then all the examples will need updating accordingly.

Note The supplied JCL member ISZRDEL4 (in library .SISZCONF) can be used to delete the ISZCLASS class from the dynamic CDT.

- 2 Define the resources to the RACF class ISZCLASS and permissions to the RACF groups.

Tailor to your installation standards and submit the supplied JCL member ISZRDEF5 (in library .SISZCONF). The following resources will be defined in the RACF class ISZCLASS and the corresponding RACF groups defined and given READ access:

Authorization level resources:

ISZ.AUTH.9	ISZAAAAA group
ISZ.AUTH.8	ISZBBBBB group
ISZ.AUTH.7	ISZCCCCC group
ISZ.AUTH.6	ISZDDDDD group
ISZ.AUTH.5	ISZEEEEEE group
ISZ.AUTH.4	ISZFFFFF group
ISZ.AUTH.3	ISZGGGGG group
ISZ.AUTH.2	ISZHHHHH group
ISZ.AUTH.1	ISZIIIII group

OLA class resources:

ISZ.OLA.AD	ISZCCCCC group
ISZ.OLA.IM	ISZAAAAAA and ISZBBBBBB groups
ISZ.OLA.LA	ISZDDDDD and ISZEEEEEE groups
ISZ.OLA.NO	ISZIIIII group
ISZ.OLA.SU	ISZFFFFFF group
ISZ.OLA.US	ISZGGGGG and ISZHHHHH groups

- 3** Update the SECURITY sub-parameters on the SYSTEM statement as follows:

```
AUTHCLASSNAME = ISZCLASS
```

```
AUTHRESNAME = ISZ.AUTH.
```

```
OLARESNAME = ISZ.OLA.
```

Note The trailing dot must be present on the AUTHRESNAME and OLARESNAME settings.

See *Technical Reference* manual for more details on these SECURITY sub-parameters.

How do I check a user is authorized to access the Session Manager ACB?

To enable an ESM to be used to control access to the Session Manager ACB, appropriate resources must be defined in the ESM and users given access to these resources. The example below illustrates how to define the resources and user's access levels in a RACF environment using the supplied members ISZRDEF6 and ISZRDEF7 (in library .SISZCONF).

The class and resource names to be used when querying the ESM are configured by the SIGNONCLASS, the SIGNONRESNAME and the SIGNONACCESS sub-parameters on the SECURITY parameter on the SYSTEM statement (see *Technical Reference* manual).

It is by configuring the SIGNONCLASS that indicates to the supplied E21exit that Session Manager should check for ACB authorization within the ESM.

Example of setting a user's access to the Session Manager' ACB

The example below assumes that 'Example of defining users and their passwords to RACF' (see page 294), defining the users and exit points, has already been applied and that the users, RACF groups and E21 exit have already been defined within RACF and Session Manager.

This example will define the RACF resources and their group permissions. It will also define the Session Manager Security settings.

The following RACF groups and their users will have access to Session Manager' ACB and therefore will be allowed to sign-on:

ISZAAAAA group	user AAAAA
ISZBBBBB group	user BBBBB
ISZCCCCC group	user CCCCC
ISZDDDDD group	user DDDDD
ISZEEEEEE group	user EEEEE

ISZAAAAA group	user AAAAA
ISZFFFFF group	user FFFFF
ISZGGGGG group	user GGGGG
ISZHHHHH group	user HHHHH
ISZIIIII group	user IIIII

All other RACF groups and their users will be revoked when they attempt to sign-on.

Although the following steps will work as supplied most sites will have to modify them to meet their own site standards, especially their RACF security standards.

- 1 Define the ISZAPPL class to the dynamic CDT in RACF or use the existing RACF APPL class.

We recommend that you use the existing RACF APPL class.

To use an alternative RACF class you should tailor to your installation standards and submit the supplied JCL member ISZRDEF6 (in library .SISZCONF). The supplied JCL member ISZRDEF6 will define an ISZAPPL class to the dynamic CDT within RACF. The class that you have selected will be defined to RACF. The POSIT value must be unique - please consult with your RACF systems programmer. If your installation has already defined a dynamic CDT then remove the SETROPTS CLASSACT(CDT) RACLIST(CDT) command.

Note The supplied JCL member ISZRDEL6 (in library .SISZCONF) can be used to delete the ISZAPPL class from the dynamic CDT.

- 2 Define the resources to the RACF class APPL and permissions to the RACF groups.
- 3 Tailor to your installation standards and submit the supplied JCL member ISZRDEF7 (in library .SISZCONF). The following resource will be defined in the RACF class APPL and the corresponding RACF groups given READ access:

```
ISZSMGR    ISZAAAAA, ISZBBBBB, ISZCCCCC, ISZDDDDD,
           ISZEEEE, ISZFFFFF, ISZGGGGG, ISZHHHHH, ISZIIIII
           groups
```

APPL is the specified RACF class in ISZRDEF7. Change ISZRDEF7 to define the RACF class that you intend to use.

ISZSMGR is the default Session Manager ACB. Change ISZRDEF7 to define the ACB that you intend to use as your main Session Manager ACB.

- 4 Update the SECURITY sub-parameters on the SYSTEM statement as follows:

```
SIGNONCLASS = APPL (or the RACF class defined in your ISZRDEF6 job)
SIGNONACCESS = NO (the default)
```

When using the APPL class the SIGNONRESNAME sub-parameter is not required. The SIGNONRESNAME sub-parameter could be used if you are using an alternative RACF class and require a high level qualifier for the APPL being used.

For example:

```
SIGNONRESNAME = ISZ.AC.B.
```

Note In this example there is a trailing dot on the SIGNONRESNAME setting.

See *Technical Reference* manual for more details on these SECURITY sub-parameters.

How do I check a terminal is authorized to access Session Manager?

To enable an ESM to be used to control which terminal can access Session Manager appropriate resources must be defined in the ESM and users given access to these resources. The example below illustrates how to define the resources and user's access levels in a RACF environment using the supplied members ISZRDEF8 and ISZRDEF9 (in library .SISZCONF).

The class and resource names to be used when querying the ESM are configured by the TERMINALCLASS, the TERMINALRESNAME and the TERMINALACCESS sub-parameters on the SECURITY parameter on the SYSTEM statement (see *Technical Reference* manual).

It is by configuring the TERMINALCLASS that indicates to the supplied E21exit that Session Manager should check for TERMINAL authorization within the ESM.

Example of setting a user's terminal access to Session Manager

The example below assumes that 'Example of defining users and their passwords to RACF' (see page 294), defining the users and exit points, has already been applied and that the users, RACF groups and E21 exit have already been defined within RACF and Session Manager.

This example will define the RACF resources and their group permissions. It will also define the Session Manager Security settings.

The following RACF groups and their users will have access to the corresponding terminals and therefore will be allowed to sign-on via these terminals:

ISZAAAAA group	user AAAAA	terminal name S09LU10
ISZBBBBB group	user BBBBB	terminal name S09LU11
ISZCCCCC group	user CCCCC	terminal name S09LU12
ISZDDDDD group	user DDDDD	terminal name S09LU13
ISZEEEEEE group	user EEEEE	terminal name S09LU14
ISZFFFFFF group	user FFFFF	terminal name S09LU15
ISZGGGGG group	user GGGGG	terminal name S09LU16
ISZHHHHH group	user HHHHH	terminal name S09LU17
ISZIIIII group	user IIIII	terminal name S09LU18

All other RACF groups and their users will be revoked when they attempt to sign-on via other terminals.

Although the following steps will work as supplied most sites will have to modify them to meet their own site standards, especially their RACF security standards.

- 1 Define the ISZTERM class to the dynamic CDT in RACF or use the existing RACF TERMINAL class.

We recommend that you use the existing RACF TERMINAL class.

To use an alternative RACF class you should tailor to your installation standards and submit the supplied JCL member ISZRDEF8 (in library .SISZCONF). The supplied JCL member ISZRDEF8 will define an ISZTERM class to the dynamic CDT within RACF. The class that you have selected will be defined to RACF. The POSIT value must be unique - please consult with your RACF systems programmer. If your installation has already defined a dynamic CDT then remove the SETROPTS CLASSACT(CDT) RACLIST(CDT) command.

Note The supplied JCL member ISZRDEL8 (in library .SISZCONF) can be used to delete the ISZTERM class from the dynamic CDT.

- 2 Define the resources to the RACF class TERMINAL and permissions to the RACF groups.
- 3 Tailor to your installation standards and submit the supplied JCL member ISZRDEF9 (in library .SISZCONF). The following resources will be defined in the RACF class TERMINAL and the corresponding RACF groups given READ access:

S09LU10	ISZAAAAA,
S09LU11	ISZBBBBB
S09LU12	ISZCCCCC
S09LU13	ISZDDDDD
S09LU14	ISZEEEEEE
S09LU15	ISZFFFFFF
S09LU16	ISZGGGGG
S09LU17	ISZHHHHH
S09LU18	ISZIIIII

TERMINAL is the specified RACF class in ISZRDEF9. Change ISZRDEF9 to define the RACF class that you intend to use.

S09LU10 to S09LU18 are just examples of Session Manager terminal names and therefore you should change ISZRDEF9 to define the terminal names that you intend to use.

- 4 Update the SECURITY sub-parameters on the SYSTEM statement as follows:

TERMINALCLASS = TERMINAL (or the RACF class defined in your ISZRDEF8 job)

TERMINALACCESS = NO (the default)

When using the TERMINAL class the TERMINALRESNAME sub-parameter is not required. The TERMINALRESNAME sub-parameter could be used if you are using an alternative RACF class and require a high level qualifier for the terminal being used.

For example:

TERMINALRESNAME = ISZ.TERM.

Note In this example there is a trailing dot on the TERMINALRESNAME setting.

See the SECURITY parameter on the SYSTEM statement in the *Technical Reference* manual for more details on these sub-parameters.

How do I control access to a Session Manager ACB not defined to RACF?

Access is controlled by the SIGNONACCESS sub-parameter on the SECURITY parameter on the SYSTEM statement.

If the ESM can not determine if a user would have access to the generated resource name (a combination of the setting in

SIGNONRESNAME and the Session Manager ACB name) then a combination of the SIGNONACCESS parameter and your RACF PROTECT ALL setting will determine if access is granted.

If RACF PROTECT ALL is active any undefined resources will return a SAF return code of 8 - access denied.

When using the standard supplied E21 exit the user is not granted access.

If RACF PROTECT ALL is not active any undefined resources will return a SAF return code of 4,4 - RACF can not make a decision. When using the standard supplied E21 exit a SIGNONACCESS setting of Yes or ON will allow the user to be signed on. A SIGNONACCESS setting of No or OFF will cause the sign on to be revoked.

See the SECURITY parameter on the SYSTEM statement in the *Technical Reference* manual for more details on the SIGNONACCESS sub-parameter.

How do I control access by a terminal not defined to RACF?

Access is controlled by the TERMINALACCESS sub-parameter on the SECURITY parameter on the SYSTEM statement.

If the ESM can not determine if a user would have access to the generated resource name (a combination of the setting in

TERMINALRESNAME and the terminal name) then a combination of the TERMINALACCESS sub-parameter and your RACF PROTECT ALL setting will determine if access is granted.

If RACF PROTECT ALL is active any undefined resources will return a SAF return code of 8 - access denied.

When using the standard supplied E21 exit the user is not granted access.

If RACF PROTECT ALL is not active any undefined resources will return a SAF return code of 4,4 - RACF can not make a decision.

When using the standard supplied E21 exit a TERMINALACCESS setting of Yes or ON will allow the user to be signed on. A TERMINALACCESS setting of No or OFF will cause the sign on to be revoked.

See the SECURITY parameter on the SYSTEM statement in the *Technical Reference* manual for more details on the TERMINALACCESS sub-parameter.

If I modify our RACF definitions how do I update Session Manager?

A Session Manager command script, SECFRESH, forces the E21 user exit to 're-RACLIST' any security classes that have been specified on the SECURITY sub-parameters on the SYSTEM statement.

Issue the SECFRESH command within your Session Manager.

Although this command will update the RACF definitions within your Session Manager they will only apply to users signing in to Session Manager. Existing signed on users must sign off and sign on again to utilize the new RACF definitions.

Are there any considerations for Top Secret installations?

If using the standard supplied E21 exit and you want to see the Top Secret messages after an attempted sign-on then you will need to change the E21 exit and re-assemble and link the new version. Using the supplied JCL member ISZEASM (in library .SISZCONF) follow the instructions at the top of the member and change to assemble and link the E21 SAF Signon exit. Tailor to your installation standards and submit the JCL. Ensure that the exit is assembled and linked into a library that is included in your Session Manager JCL. If using the supplied E21 exit script ISZE21PH, then the script variable `t_tss_emsg` must be set to 'Y'. Uncomment the LET statement at the top of the supplied ISZE21PH to enable this facility.

Implementing Dynamic Menus

Session Manager menus for particular users can be configured dynamically from access rules specified using the Installation's External Security Manager (ESM).

How is the Dynamic Menu facility provided?

When a user signs on to Session Manager he will have one or more PROFILES defined which will contain a list of all their potentially available Session Manager sessions. The DYNMDROPSESSION sub-parameter on the SECURITY parameter (see the SYSTEM statement in the *Technical Reference* manual) determines whether to action the HIDE or DROP_SESSION value. The E22 user exit will set HIDE or DROP_SESSION accordingly and will ignore any configuration settings for these parameters.

The Dynamic Menu facility is provided using an E22 (signon completion) user exit. The supplied user exit, ISZE22DM (in library .SISZCONF), performs security checks against each session and builds a list of Session Manager sessions that are available to the user.

Note The E21 (signon validation) user exit (see User Authentication section above and 'Signon validation exit point (E21)' on page 217 for more details) is still required. The purpose of this user exit is to authenticate users and set some of their Session Manager capabilities using an Installation's external security system.

Sample user exit

If you require your own, non-standard, exit routine please refer to the *User and Administrator* manual which contains a full description of how such an exit routine should be used.

The following sample user exit is supplied on the distribution media:

ISZE22DM External Security Manager (ESM) exit

It is supplied in source form and in executable form. If required, it can be modified and then should be assembled and linked in the normal manner.

The ISZE21SF or ISZE21PH exit is required and should be installed and configured as described above in the User Authentication section.

Storage use and performance

The DYNMDROPSESSION sub-parameter on the SECURITY parameter (see the SYSTEM statement in the *Technical Reference* manual) determines whether to action the common session parameter's HIDE or DROP_SESSION value. If an installation uses the HIDE common session parameter to stop the display of applications that the user doesn't have authority to access, the user will still have storage associated with these sessions, even if they are hidden. These unused sessions will also be scanned each time the user presses Enter, to verify that the user has not entered a TRANSID referring to them, thus potentially creating both a storage and CPU overhead.

If an installation uses the `DROP_SESSION` common session parameter to stop the display of applications that the user doesn't have authority to access, the storage associated with these sessions will be deleted and the session will be dropped. These unused sessions will therefore not be scanned each time the user presses Enter, to verify that the user has not entered a `TRANSID` referring to them. This optimizes the use of storage and potentially improves performance. However you will not be able to autoselect or autostart these dropped sessions.

FAQs

How does the E22 user exit establish a user's sessions?

The E22 user exit establishes the number of sessions available to the user as follows:

Each Session Manager session has an APPL and/or VTAM APPLID. This value is checked against an ESM resource profiles in a given ESM Class and if the user has READ access then the session will be made available.

For sessions that specify an APPL and/or VTAM APPLID to which the user has READ access, the session will be 'unhidden' by modifying the `s_hidden` or `s_dropsess` variable (see the *Panels, Scripts and Variables* manual) for that session. Only these sessions will then appear on the user's Session Manager menu.

For example, if the supplied sample is used unchanged, for RACF the resource profile APPL will be checked in the class specified by the `DYNMCLASS` sub-parameter under SECURITY on the SYSTEM statement. The default for this class is FACILITY.

If a `DYMNRESNM` sub-parameter has been specified with a value `dynamic_menu_resname`. under SECURITY on the SYSTEM statement then the resource profile `dynamic_menu_resname.APPL` will be checked.

The actual number of sessions available to the user is likely to be less than the maximum number of expected sessions.

Other options to be considered on the SECURITY parameter of the SYSTEM statement are:

- The `DYMNALOG` sub-parameter determines whether an error message is recorded in the audit log when the ESM cannot determine if a user has access to a session.
- The `DYMNAUTSTHID` sub-parameter determines whether sessions hidden from a user may be autostarted or autoselected.
- The `DYMNHIDE` sub-parameter determines whether sessions are hidden or dropped from users who cannot be granted access by the ESM.
- The `DYMNDRPSESSION` sub-parameter determines whether to action the common session parameter's `HIDE` or `DROP_SESSION` value.
- The `DYMNLOGMAX` sub-parameter places a limit on the number of `DYMNALOG` messages recorded in the audit log.
- The `DYMNTYPE` sub-parameter determines whether the VTAM APPLID or APPL should be used when checking a user's access authorization with the ESM.

For further information on these sub-parameters, see the questions below and the SECURITY parameter on the SYSTEM statement in the *Technical Reference* manual.

How do I configure Session Manager for Dynamic Menus?

Users can be configured with multiple PROFILES, up to a maximum of eighteen. This is achieved by assigning multiple PROF parameters to the USER statement or by configuring multiple DEFPROFILE parameters on the SYSTEM statement and setting the OPTION parameter MDPROF to M. These PROFILES will define all the sessions that the user *may* have access to. The example below will use the PROFILE as defined on the DEFPROFILE parameter on the SYSTEM statement. This approach allows for all the installation's sessions to be defined in one area whilst utilizing RACF to determine which sessions any given user can access. Therefore access control to the sessions is controlled solely by RACF.

The example below assumes that 'Example of defining users and their passwords to RACF' (see page 294), defining the users and exit points, has already been applied and that the users, RACF groups and E21 exit have already been defined within RACF and Session Manager.

- 1 Add supplied PROFILE and APPLs to your Session Manager configuration.

If not already incorporated then ensure that the supplied members ISZCRPRF and ISZCRAPL (in library .SISZCONF) are incorporated into your Session Manager system.

For Classic systems:

Add COPY ISZCRPRF and COPY ISZCRAPL to your Session Manager configuration.

For OLA systems:

Tailor to your installation standards and submit the supplied JCL member ISZRDF10 (in library .SISZCONF). This job will use the ISZLOJOB facility to add the PROFILES and APPLs to your OLA PROFILE and APPL datasets.

- 2 Either ensure the supplied ISZE22DM is in your Session Manager library or, if required, modify and assemble the supplied E22 user exit ISZE22DM.
- 3 If you have modified the E22 exit, assemble and link it using the supplied JCL member ISZEASM. Follow the instructions at the top of the member for changing it to assemble and link the E22 SAF Sign-on completion exit. Tailor to your installation standards and submit the JCL. Ensure that the exit is assembled and linked into a library that is included in your Session Manager JCL.
- 4 Update the SECURITY sub-parameters on the SYSTEM statement as follows:

DYNMCLASS = FACILITY (the default)

DYNMRESNM = ISZ.APPL.

Note The trailing dot must be present on the DYNMRESNM setting.

See *Technical Reference* manual for more details on these SECURITY sub-parameters.

- 5 Update the DEFPROFILE parameter on the SYSTEM statement as follows:

DEFPROFILE = ISZCRPRF

See *Technical Reference* manual for more details on the DEFPROFILE parameter.

How do I configure RACF for Dynamic Menus?

To enable an ESM to be used to control access to Session Manager sessions, appropriate resources must be defined in the ESM and users given access to these resources. The example below illustrates how to define the ESM class, resources and user's access levels in a RACF environment using the supplied members ISZRDF11 and ISZRDF12 (in library .SISZCONF).

The class and resource names to be used when querying the ESM are configured by the DYNMCLASS and DYNMRESNM sub-parameters on the SECURITY parameter on the SYSTEM statement (see *Technical Reference* manual).

An example to create a user's session list.

The example below assumes that 'Example of defining users and their passwords to RACF' (see page 294), defining the users and exit points, has already been applied and that the users, RACF groups and E21 exit have already been defined within RACF and Session Manager.

This example will define the RACF class to be used and define the RACF resources and their group permissions.

After signing on to Session Manager the users will have access to the following sessions:

ISZAAAAA	user AAAAA	Session APPL AAAAA
ISZBBBBB	user BBBBB	Session APPL BBBBB
ISZCCCCC	user CCCCC	Session APPL CCCCC
ISZDDDDD	user DDDDD	Session APPL DDDDD
ISZEEEEEE	user EEEEE	Session APPL EEEEE
ISZFFFFF	user FFFFF	Session APPL FFFFF
ISZGGGGG	user GGGGG	Session APPL GGGGG
ISZHHHHH	user HHHHH	Session APPL HHHHH
ISZIIIII	user IIIII	Session APPL IIIII

Although the following steps will work as supplied most sites will have to modify them to meet their own site standards, especially their RACF security standards.

- 1 Define the ISZSESS class to the dynamic CDT in RACF or use the existing RACF FACILITY class.

We recommend that you use the existing RACF FACILITY class.

To use an alternative RACF class you should tailor to your installation standards and submit the supplied JCL member ISZRDF11 (in library .SISZCONF). The supplied JCL member ISZRDF11 will define an ISZSESS class to the dynamic CDT in RACF. The class that you have selected will be defined to RACF. The POSIT value must be unique - please consult with your RACF systems programmer.

Note The supplied JCL member ISZRDL11 (in library .SISZCONF) can be used to delete the ISZSESS class from the dynamic CDT.

- 2 Define the resources to the RACF class FACILITY and permissions to the RACF groups.

- 3 Tailor to your installation standards and submit the supplied JCL member ISZRDF12 (in library .SISZCONF). The following resources will be defined in the RACF class FACILITY and the corresponding RACF groups given READ access:

ISZ.APPL.AAAAA	ISZAAAAA group
ISZ.APPL.BBBBB	ISZBBBBB group
ISZ.APPL.CCCCC	ISZCCCCC group
ISZ.APPL.DDDDD	ISZDDDDD group
ISZ.APPL.EEEEE	ISZEEEEEE group
ISZ.APPL.FFFFF	ISZFFFFFF group
ISZ.APPL.GGGGG	ISZGGGGG group
ISZ.APPL.HHHHH	ISZHHHHH group
ISZ.APPL.IIIII	ISZIIIII group

Can a user continue to use a dynamic menu?

The user can continue to use the dynamic menu, with security checks being performed against their Profiles and a list of available sessions being created each time the user logs on. This allows new sessions (pre-defined on the default PROFILE) to be displayed for a user by changing the user's external security rules.

Can a 'static' USER definition be built from a dynamic menu?

Providing that OLA is used to administer Session Manager (see the Online and Batch Administration manual), a special session script, ISZE22SM (in library .SISZCONF), can be defined in the default PROFILE that is run automatically at signon. This script, which runs after the E22 user exit, can be used to create a 'static' USER definition with just the number of sessions currently available to the user.

It will also change the user's PROFILE to ISZE22SM. This dummy PROFILE, ISZE22SM, must exist in the OLA configuration. A minor change is also required to the ISZE22DM exit so that it ignores users with this profile. Instructions are at the top of the supplied ISZE22DM exit.

See 'Example configuration statements' on page 312.

How do I update sessions for a 'static' USER definition built from a dynamic menu?

The user can change this USER definition using OLA.

Alternatively by deleting this USER definition it will be rebuilt when the user next signs on to Session Manager.

Can error messages in the audit log be turned on or off?

The DYNMLOG sub-parameter on the SECURITY parameter of the SYSTEM statement determines if message 4028 will be recorded in the audit log whenever the ESM cannot determine whether a user should have access to a session. A setting of Yes or ON will cause the message to be recorded. A setting of No or OFF will prevent a message from being recorded.

See the SECURITY parameter on the SYSTEM statement in the *Technical Reference* manual for more details on the DYNMLOG sub-parameter.

Can the number of error messages in the audit log be limited?

The DYNMLOGMAX sub-parameter on the SECURITY parameter of the SYSTEM statement sets a limit on the number of DYNMLOG 4028 messages written to the audit log during each user sign-on. Any value from 0 to 9999 may be specified.

See the SECURITY parameter on the SYSTEM statement in the *Technical Reference* manual for more details on the DYNMLOGMAX sub-parameter.

Can the APPL name or VTAM applid be used as part of the ESM resource name?

The DYNMTYPE sub-parameter on the SECURITY parameter of the SYSTEM statement determines whether the APPL or VTAM APPLID should be used when checking a user's access authorization with the ESM. A DYNMTYPE of APPL will cause the APPL name to be used. A DYNMTYPE of VTAMAPPL will cause the VTAM APPLID to be used.

See the SECURITY parameter on the SYSTEM statement in the *Technical Reference* manual for more details on the DYNMTYPE sub-parameter.

Can hidden or dropped sessions be autostarted or autoselected?

The DYNMAUTSTHID sub-parameter on the SECURITY parameter of the SYSTEM statement determines whether sessions hidden from a user may be autostarted or autoselected. A setting of Yes or ON means that hidden sessions can be autostarted or autoselected. A setting of No or OFF means that hidden sessions cannot be autostarted or autoselected.

The DYNMDROPSSESSION sub-parameter on the SECURITY parameter determines whether to action the common session parameter's HIDE or DROP_SESSION value. Dropped sessions can not be autostarted or autoselected.

See the SECURITY parameter on the SYSTEM statement in the *Technical Reference* manual for more details on the DYNMAUTSTHID and DYNMDROPSSESSION sub-parameters.

What determines if a session is hidden or dropped and what is the difference?

The DYNMDROPSSESSION sub-parameter on the SECURITY parameter determines whether the E22 user exit will set the common session parameter's HIDE or DROP_SESSION value.

A setting of Yes or ON means that any sessions to which the user does not have access will be dropped by the E22 user exit, by setting DROP_SESSION to Y. A setting of No or OFF means that any sessions to which the user does not have access will be hidden by the E22 user exit, by setting HIDE to Y.

Dropped sessions free up more storage and potentially improve performance but they can not be autostarted, autoselected or used as part of the Application Builder process.

See the SECURITY parameter on the SYSTEM statement in the *Technical Reference* manual for more details on the DYNMDROPSSESSION sub-parameter.

How do I control access to sessions not defined to RACF?

Access is controlled by the DYNMHIDE sub-parameter on the SECURITY parameter on the SYSTEM statement.

If the ESM can not determine if a user would have access to a session then a combination of the DYNMHIDE sub-parameter and your RACF PROTECT ALL setting will determine if access is granted.

If RACF PROTECT ALL is active any undefined resources will return a SAF return code of 8 - access denied.

When using the standard supplied E22 exit the session will either be hidden or dropped, depending on the DYNMDropSession parameter setting.

If RACF PROTECT ALL is not active any undefined resources will return a SAF return code of 4,4 - RACF can not make a decision. When using the standard supplied E22 exit a setting of Yes or ON will cause these sessions to be either hidden or dropped, depending on the DYNMDropSession parameter setting. A setting of No or OFF will cause the sessions to be visible.

The E22 exit settings will override any configuration common session HIDE or DROP_SESSION settings.

See the SECURITY parameter on the SYSTEM statement in the *Technical Reference* manual for more details on the DYNMHIDE and DYNMDROPSSESSION sub-parameters.

If I modify our RACF definitions how do I update Session Manager?

A Session Manager command script, SECFRESH, forces the E22 user exit to 're-RACLIST' any security classes that have been specified on the SECURITY parameters on the SYSTEM statement.

Issue the SECFRESH command within your Session Manager.

Although this command will update the RACF definitions within your Session Manager they will only apply to users signing in to Session Manager. Existing signed on users must sign off and sign on again to utilize the new RACF definitions.

Dynamic Menu summary

Session Manager' dynamic menus facility enables menus to be built dynamically for particular users, based on definitions specified using the Installation's ESM. This facility is provided using an E22 (signon completion) user exit ISZE22DM. This exit builds a list of sessions available to a particular user based on definitions specified using the Installation's ESM. Providing that OLA is being used to administer the Session Manager system, a special session script, ISZE22SM, can be defined in the default PROFILE that is run automatically at signon. This script, which runs after the E22 user exit, can be used to create a 'static' USER definition with just the number of sessions currently available to the user.

Example configuration statements

Here are some simple examples of the Session Manager configuration statements that are required to set up dynamic menus:

```

SYSTEM
DEFPROFILE iszcrprf

PROFILE iszcrprf
MENU menu2
HCREQUEST /H
KEY PF01
APPLID aaaaa
HIDE yes
KEY PF02
APPLID bbbbb
HIDE yes

```

If a 'static' USER definition is required after first log on, configuration statements like these should be added to the default PROFILE:

```

SESSION 099
AUTOSTART yes
INITSCRIPT isze22sm
HIDE yes

```

Although the special session script, ISZE22SM, can be tailored to the Installation's requirements, the version as shipped will create a user with a new PROFILE (to prevent re-running of the script) that contains only non-hidden sessions available to the user. For example:

```

USER xxxxxxxx
PROF isze22sm
MENU MENU2
HCREQUEST /H
KEY PF02
APPLID bbbbb

```

The dummy PROFILE, ISZE22SM, must exist in the OLA configuration. A minor change is also required to the ISZE22DM exit so that it ignores users with this profile. Instructions are at the top of the supplied ISZE22DM exit.

Subsequently, the user can change this USER definition using the Online Administration facility (OLA). Alternatively, by deleting this USER definition it will be rebuilt when the user next signs on to Session Manager.

The PROFILE definition can use the DROP_SESSION parameter rather than the HIDE parameter. For example:

```

SYSTEM
DEFPROFILE iszcrprf

PROFILE iszcrprf
MENU menu2
HCREQUEST /H
KEY PF01
APPLID aaaaa
DROP_SESSION yes
KEY PF02
APPLID bbbbb
DROP_SESSION yes

```

If all sessions in the PROFILE are to be subject to DROP_SESSION checking, the DROP_SESSION parameter need only be specified in the PROFILE header. For example:

```
SYSTEM
DEFPROFILE iszcrprf

PROFILE iszcrprf
MENU menu2
HCREQUEST /H
DROP_SESSION yes
KEY PF01
APPLID aaaaa
KEY PF02
APPLID bbbbb
```

CHAPTER 10

PassTickets and Certificate Express Logon

It is possible to use PassTickets – encrypted, time-limited tokens – rather than sending unencrypted passwords when logging onto applications through IBM Session Manager for z/OS.

PassTickets can be generated automatically by Session Manager and passed to the application, so that the user can be authorized without needing to enter a new PassTicket for every application logged onto using Session Manager.

Certificate Express Logon, formerly known as the Express Logon Feature (ELF), allows users running a 3270 client session to log on to a host system without having to enter their credentials; that is, their userid and password.

PassTickets

PassTicket support is provided by a number of sample Session Manager exits and scripts, which are shown in the tables that follow. Different scripts and exits are required depending on whether Session Manager is used in its usual 'Relay' mode, where more than one application can be accessed simultaneously, or in its 'Front-end' mode, where only one application can be used. See 'Using Session Manager as a 'front-end'' on page 425 for more information.

The exits in the following tables are included as individual members of the shipped configuration; the scripts are supplied in a single member – ISZSPSTK (in library .SISZCONF).

Note The exits and scripts provided are samples and should, if necessary, be tailored to meet the requirements of your Installation.

The variable in which the generated PassTicket is to be passed to the application must be specified on the DATA parameter of the appropriate APPL statement. For detailed information on the APPL statement, refer to the *Technical Reference*.

Using Session Manager as a 'Front-end'

The following script and exit are required to implement PassTicket support:

Name	Type	Description
E09PASS	Script	Initialization script for the set up of parameters and the call of Exit 9. Set the common session parameter INITSCRIPT to call this script.
ISZEPTKT	Exit	Exit 9, driven by E09PASS. Generates and returns a PassTicket for the application.

Using Session Manager in 'Relay' mode'

The following scripts and exit are required to implement PassTicket support:

Name	Type	Description
E31INITS	Script	Initialization script to move data to user variables for subsequent use by the E31 exit and E31 script. Set the common session parameter INITSCRIPT to call this script.
EXIT31	Script	Exit script to set up the PassTicket token variable for the main Exit 31, and to either set the PassTicket variable to the password or to call the exit to generate a PassTicket. The OPTION statement for the E31 exit should be modified to include the 'S' script switch so that this script is called.
ISZE31PT	Exit	Exit 31, driven by EXIT31. Generates and returns a PassTicket for the application.

Name	Type	Description
TSOSCRPT	Script	Start script for logon to a TSO session using the PassTicket variable. Set the common session parameter STARTSCRIPT to call this script.

Session Manager parameters for PassTicket processing

PassTicket processing uses two common session parameters, PSTKApp1 and PSTKUser. For details see the Common Parameters chapter in the *Technical Reference* manual.

Certificate Express Logon

Certificate Express Logon, formerly known as Express Logon Feature (ELF), allows users running a 3270 client session to log on to a host system without having to enter their credentials; that is, their userid and password. The host session must be configured for SSL and client authentication. This means the client must have a valid client certificate. The SSL connection must be made to one of the supported TN3270 servers.

The following documentation details how to configure SSL and TN3270:

z/OS V1R10.0 Communications Server IP Configuration Reference

z/OS V1R10.0 Communications Server IP Configuration Guide

When verifying a user against your External Security Manager, if you do not want to default to verifying access against the Session Manager ACB name then the Session Manager SYSTEM parameter GENRESNAME must be set to the resource name against which you wish to verify; for example, the VTAM generic resource name.

In order to use Certificate Express Logon, you must configure the telnet servers and the z/OS system that you are accessing.

- 1 Start up a 3270 session to the host system you wish to connect to.
- 2 Make sure you are at the start screen of the host application before you start recording. This should be the logon screen or the one before it. **Do not log on.**
- 3 Record a logon macro. Click **Record Macro** on the Macro toolbar. Type a name and description for the macro. Use a name that will help you to recognize it as an Express Logon macro.
- 4 Select the **Express Logon** feature.
- 5 Enter the application ID. This is the host access control facility's application ID.
- 6 Continue through the Certificate Express Logon macro recording. Click **Help** if you need more information. If you enter the wrong data while recording a macro, you cannot go back to make corrections. You can, however, record over the existing macro or edit the macro code to make changes.
- 7 Click **Stop Recording** when you have completed the recording of all the screens.
- 8 Enable SSL and client authentication. Right-click the session icon, select **Properties**, and then click the **Security** tab.
- 9 Test the macro. Click **Play Macro** on the Macro toolbar. This should sign you onto Session Manager.

CHAPTER 11

The Shared Userid facility

The Shared Userid facility enables multiple users to sign on with the same userid.

The subjects covered in this chapter are:

- 'Overview' on page 320
- 'Operational considerations' on page 321
- 'Summary' on page 323.

Overview

The Shared User facility allows multiple users to share the same userid for signing on.

This is achieved by doing one of the following:

- specifying `SHARE Yes` as a common enduser parameter
- specifying `SHAREDISC Yes` as a common enduser parameter
- specifying `SHARESESS Yes` as a common enduser parameter
- setting the Multiple Users flag to 'M' in the E21 Exit (or the variable `ec21_ruser` in the E21 exit script).

A unique, or fully qualified, identifier is created internally for each user by appending a 'user qualifier' to the signon userid. Depending on the value of parameter `MULTUSER` on the `SYSTEM` statement (see the *Technical Reference*), the 'user qualifier' can be created from the last four characters of the terminal LU name, or it is an eight-digit number in the range 1-99999999. The fully qualified identifier is used internally only and is held in the user associated variable `t_user_qual`; the userid visible to the user on the screen and in messages remains the one with which they signed on.

The Shared Userid facility can be controlled by the `SHARE`, `SHAREDISC` and `SHARESESS` common enduser parameters, or by setting a flag in the Signon Validation Exit Point (E21) of the Session Manager User Exit.

Using the `SHARE` common enduser parameter

To define a user as a 'shared user', you can specify the common enduser parameter `SHARE Yes` (or `SHARE ON`). For details, see the *Technical Reference*.

Using the `SHAREDISC` common enduser parameter

An alternative to `SHARE`, the `SHAREDISC` parameter enables one userid to be used to log on to multiple terminals while sharing the same sessions and menu. If a user on another terminal takes control of a shared session, the initial user is logged off or disconnected. For details, see the *Technical Reference*.

Using the `SHARESESS` common enduser parameter

An alternative to `SHARE` and `SHAREDISC`, the `SHARESESS` parameter also enables one userid to be used to log on to multiple terminals, but its effect on the initial user when a session is taken over differs. If a user on another terminal takes control of a shared session the initial user loses control of the session and receives a message to that effect. For details, see the *Technical Reference*.

Using the E21 Session Manager User Exit

To enable the facility, the E21 script variable `ec21_ruser` must be set to 'M' by the Exit when the return code is zero. For a description of the E21 Exit, see 'Signon validation exit point (E21)' on page 217. If running the supplied E21 exit script `ISZE21PH`, then the script must be amended as detailed above.

Operational considerations

Signing on

If a user attempts to sign on at a terminal then, depending on the value of parameter `MULTUSER` on the `SYSTEM` statement (see the *Technical Reference*), either a ‘user qualifier’ is created from the last four characters of the terminal LU name, or it is an eight-digit number in the range 1-99999999. If this ‘user qualifier’ does not give a unique, fully-qualified name for active users, the user will receive the message:

```
ISZ0248I  userid already signed on at LU termid
```

and will be unable to sign on. If this occurs the user should enter the `LOGOFF` command then attempt to sign on at a different terminal where the ‘user qualifier’ is unique.

Reconnecting

The `RECONYTERM` parameter of the `SYSTEM` statement enables any ‘`SHARE=Y`’ user to reconnect to a disconnected user’s sessions with the same name from a different terminal. For details, see the *Technical Reference*.

Commands

These situations can arise:

- If the command:

```
STOP USER userid
```

is issued when there are multiple users signed on with `userid`, you will receive the message:

```
ISZ0521E  STOP command rejected - multiple users
```

- Certain other commands which refer to a particular user may fail when the Shared Userid facility is active, with the message:

```
ISZ0303I  USER userid not signed on
```

In these situations you can:

- 1 Issue this command:

```
QUERY USER userid
```

If multiple users are signed on with the same userid then, from the resulting display, you can distinguish which one is required by the terminal LU name.

Note Optionally, you can include either a four-character or an eight-digit `user_qualifier` as appropriate with the ‘`QUERY USER`’ command to distinguish which user is required (see the *Technical Reference*).

- 2 Reissue the command which failed, specifying the LU name of the terminal instead of the userid. For example:

```
STOP LU luname
```

Alternatively, if the command was a `STOP` command, then reissue the command with these parameters to distinguish which user is required:

```
STOP USER userid user_qualifier
```

where `user_qualifier` is either a four-character or an eight-digit ‘user qualifier’ as appropriate (see the *Technical Reference*).

- If the `WINDOWS` command is issued on a terminal using either `SHAREDISC` or `SHARESESS`, the following message will appear:


```
ISZ0518I Windows not allowed with a multi-terminal user
```
- The `RESET` command can be issued from any terminal. The session will be reset whether or not the terminal issuing the `RESET` command is the owner of the session.

MISER and PCTransfer

All `SHAREDISC` and `SHARESESS` sessions are automatically started with `MISER ERB` turned on. Having the `PCTransfer` parameter set to `Yes` (see the *Technical Reference* for details) disables `MISER`, so it is recommended that `PCTransfer Yes` is not set; instead, only enable `PCTransfer` during the period when a file is being transferred. The supplied example script `GOPTRAN`, to be used with `SAUTOSEQ`, is a possible implementation of this.

Using Session Manager as a 'Front-end'

The common session parameters `CLOSELOGOFF` and `CLOSEDISC` can be used to enable Session Manager to be used as a 'Front-end'. If `SHAREDISC` or `SHARESESS` are being used then `CLOSELOGOFF` will behave as `CLOSEDISC` for the primary user (the first terminal logged on with the userid), but for secondary users `CLOSELOGOFF` will behave as usual.

For more information, see 'Using Session Manager as a 'front-end'' on page 425.

Using IDLELOGOFF

The common enduser parameter `IDLELOGOFF` enables a time interval to be set after which users are logged out if they are inactive. If the Shared Userid facility is implemented using `SHAREDISC` or `SHARESESS`, the operation of `IDLELOGOFF` alters slightly:

- If the logoff interval has expired for the primary user then they will be logged off if no secondary users are connected. If any secondary users are connected then the primary user will be disconnected rather than logged off.
- If the logoff interval has expired for a secondary user then only that secondary user will be logged off if any primary or other secondary user is connected. However, if primary or other secondary users are disconnected then those other users will also be logged off.

Processing requirements

Operating with `SHAREDISC` and `SHARESESS` incurs a small additional processing overhead in comparison with standard, non-shared, operation.

Summary

These parameters are used to control the Shared Userid Facility:

Parameter	Description
SHARE	To define a user as a 'shared user', you can specify the common enduser parameter <code>SHARE Yes</code> (or <code>SHARE ON</code>). For details, see the <i>Technical Reference</i> .
SHAREDISC	An alternative to <code>SHARE</code> , the <code>SHAREDISC</code> parameter enables one userid to be used to log on to multiple terminals while sharing the same sessions and menu. If a user on another terminal takes control of a shared session, the initial user is logged off or disconnected. For details, see the <i>Technical Reference</i> .
SHARESESS	An alternative to <code>SHARE</code> and <code>SHAREDISC</code> , the <code>SHARESESS</code> parameter also enables one userid to be used to log on to multiple terminals, but its effect on the initial user when a session is taken over differs. If a user on another terminal takes control of a shared session the initial user loses control of the session and receives a message to that effect. For details, see the <i>Technical Reference</i> .
ec21_ruser	Exit script variable in the E21 Exit. For details, see 'Signon validation exit point (E21)' on page 217. Must be set to 'M' when the return code from the exit is zero, to permit shared userids. The default is blank: users cannot share the same userid.

Network Data Minimiser (MISER)

The IBM Session Manager for z/OS Network Data Minimiser, otherwise known as MISER, provides the facility to reduce the amount of data traffic both to and from applications. Depending on the nature of the data streams involved, compression of up to ninety-five percent can be achieved. The significant reduction in data traffic enables better terminal response times, the use of lower baud rate lines, or more terminals per line.

MISER operates by maintaining an image in storage of the data displayed on the terminal. Sophisticated compression techniques mean that this buffer rarely exceeds 4K bytes in size, and typically runs at 1 to 2K bytes, minimizing paging overhead.

The subjects covered in this chapter are:

- 'MISER for outbound data streams' on page 326
- 'MISER for inbound data streams' on page 326
- 'Incompatibilities between MISER and non-miser operation' on page 327
- 'MISER overhead' on page 328
- 'Restrictions to the use of MISER' on page 330
- 'Data stream compression in Session Manager' on page 331
- 'Using both COMPRESS and MISER' on page 332
- 'Data compression and remote sessions' on page 333
- 'Summary' on page 335

MISER for outbound data streams

When an application issues an Erase Write data stream to a terminal, it is often re-writing completely, or partially, what is already displayed on the screen. MISER (when set to MISER OUT or Yes) compares its in-storage image of the screen with the outbound data stream, field by field. When a field is identical to that currently displayed on the terminal, the field is removed from the outbound data stream, and the outbound data stream is altered to a Write. At the end of this process, appropriate nulls are inserted into the data stream to overwrite characters on the screen which should be erased, and network data traffic is therefore minimized. If the insertion of these nulls would make the data stream longer than it was initially, the original data stream is sent to the terminal, so MISER never increases outbound network data traffic.

MISER for inbound data streams

MISER (when set to MISER IN or Yes) may also reduce traffic for inbound data streams. Some applications send all unprotected fields in a data stream with the Modified Data Tag (MDT) set on. Regardless of data being entered into the unprotected field, each such field is returned when the terminal is read. This simplifies the application program, since any given field in the inbound data stream is always in the same position. This method, however, often makes the inbound data stream longer than necessary. MISER improves this situation by setting all MDTs to off in the outbound data stream, while leaving them on in the in-storage screen image. When the inbound data stream is processed, any MDT-on fields in the in-storage screen image, which do not appear in the inbound data stream, are inserted at the correct position before the data is sent to the application. The application can find its data, and inbound network data traffic is minimized.

Inbound data traffic is also reduced where the Read Buffer (RBUF) command is commonly used. The RBUF command requests a terminal to transmit the complete contents of its buffer to the application. For a Model 5 terminal this is a minimum of 3564 bytes, and could be much more if extended data streams are in use, so this command can create a significant amount of network data traffic if used frequently. Indeed, the RBUF command is typically perceptibly slow even on locally-attached terminals. Session Manager itself makes use of the RBUF command when switching sessions, since the screen image must be saved for the time when the session is reselected. It is also often used by application programs. Since MISER has an in-storage image of the terminal buffer, Session Manager issues the Read Modified (RMOD) command to the terminal, updates the in-storage image, and builds the equivalent of a Read Buffer data stream from the image. This data stream may be sent to the application, or used internally by Session Manager.

Determining inbound savings

MISER removes the MDT on bits in the datastream as they arrive from an application in order to receive back only the data that has changed from the terminal. This means that smaller datastreams arrive in from the terminals to Session Manager than without MISER. However, as the application had the MDT bits on, it is expecting to see that field in its inbound datastreams even though it is unchanged. MISER inserts the missing fields so that the application is unaware of any change. Messages 427 and 433 provide details of these savings.

Incompatibilities between MISER and non-miser operation

Incompatibility may occur when emulating the RBUF command. This command is used when MISER ERB has been specified to request Read Buffer emulation. The command returns the entire terminal buffer, including nulls. The RMOD command does *not* return nulls, and since this is used in RBUF emulation, incompatibility exists.

This might be noticed where a terminal user has positioned the cursor away from the beginning of a field, by use of the cursor movement keys rather than the space bar, and entered data. The gap between the beginning of the field and the data may be filled with nulls. If the application performs an RBUF to the terminal, the nulls will be in the inbound data stream and the application can determine where the data was entered on the screen. With MISER RBUF emulation, it appears to the application that the data has been entered at the beginning of the field. Should the screen be redisplayed, the data in the field will appear to have moved to the left.

If this incompatibility is considered to outweigh the benefits of network data traffic reduction, then RBUF emulation may be selectively disabled by specifying the MISER options for the particular application as:

```
MISER INPUT OUTPUT
```

omitting the ERB option which requests RBUF emulation.

MISER overhead

In order to achieve a high rate of data traffic reduction, and to provide RBUF emulation, Session Manager needs to perform extra processing and therefore the use of MISER increases the CPU usage.

While every effort has been made to make the process efficient, a lot of work has to be done to maintain the in-storage image. The amount of work is dependent on the nature of the data streams which MISER is sent by the application. Since many host systems are I/O-bound it may well be that the CPU cycles are available.

In other Installations, it may be desirable to restrict MISER to those users, applications, or terminals which will receive maximum benefit. For example, an ISPF user on a locally attached terminal who switches sessions infrequently would benefit relatively little from MISER, since ISPF is quite efficient in what it writes to the screen. By contrast, a user on a 4800-baud line who switches frequently between two CICS sessions could realize major improvements in response time.

A judgement must be made as to whether the benefit is justified by the cost. To assist in this process MISER keeps statistics recording the volume of data processed and the degree of compression achieved. The statistics can be displayed using the QUERY STATS command.

QUERY STATS output for MISER

An example of responses to a QUERY STATS command is given in this section, and is followed by explanations to assist in understanding the statistics.

```
ISZ0426I MISER OUTBOUND BUFFERS =50, BYTES IN 13912, BYTES OUT 8321
ISZ0432I MISER OUTBOUND COMPRESSION 40 % SAVED
ISZ0427I MISER INBOUND BUFFERS = 28, BYTES IN 218, BYTES OUT 194
ISZ0433I MISER INBOUND COMPRESSION 11 % SAVED
ISZ0428I MISER APPLICATION READ BUFFERS = 2, BYTES = 7184
ISZ0429I MISER INTERNAL READ BUFFERS = 0, BYTES = 0
ISZ0434I COMPRESSED BUFFERS = 52, BYTES IN 20368, BYTES OUT 13926
ISZ0435I BUFFER COMPRESSION = 32 % SAVED
```

MSG426 Outbound buffers = *out_buf*
 Bytes in = *bytes_in*
 Bytes out = *bytes_out*

Variables

out_buf is the number of buffers from the application to the terminal which were processed by MISER.

bytes_in is the number of data bytes in the buffers before MISER compression.

bytes_out is the number of data bytes in the buffers after MISER outbound compression (that is, the number of bytes sent to the screen).

MSG432 Percentage saved = *perc_saved*

Variables

perc_saved is $100 - (bytes_out / (bytes_in / 100))$.

(*bytes_out* and *bytes_in* are as defined above for message MSG426.)

MSG427 Inbound buffers = *in_buf*
 Bytes in = *bytes_in*
 Bytes out = *bytes_out*

Variables

in_buf is the number of buffers from the terminal to the application which were processed by MISER.

bytes_in is the number of data bytes in the buffers before MISER compression.

bytes_out is the number of data bytes in the buffers after MISER MDT compression (that is, after MDT fields not sent to the terminal have been added).

MSG433 Percentage saved = *perc_saved*

Variables

perc_saved is $100 - (bytes_out / (bytes_in / 100))$.

(*bytes_out* and *bytes_in* are as defined above for message MSG427.)

MSG428 Application read buffers = *appread_buf*
 Bytes = *bytes*

Variables

appread_buf is the number of read buffers issued by an application which were emulated by MISER RBUF compression.

bytes is the number of bytes which would have been transmitted from the terminal had the read buffers actually been performed.

MSG429 Internal read buffers = *intread_buf*
 Bytes = *bytes*

Variables

intread_buf is the number of read buffers which would have been issued by Session Manager itself while switching between sessions which were emulated by MISER RBUF compression.

bytes is the number of bytes which would have been transmitted from the terminal had the read buffers actually been performed.

Restrictions to the use of MISER

MISER fully supports 3270 data streams found in the Outbound 3270DS Structured Field, as documented in the *IBM 3270 Data Stream Programmer's Reference*, provided they are destined for Implicit Partition Zero. Equally, non-Write Structured Field data streams are fully supported. MISER operation for a session terminates, issuing a message and transparently returning the user to the no-MISER method of session management, if either of the following conditions arise:

- The application sends a Create Partition structured field to the terminal.
- The application sends a graphics data stream to the terminal, for example, an Object Image Outbound Structured Field.

Should this occur, MISER is temporarily suspended, and is restarted automatically when the application sends an erase-type write to the terminal.

Data stream compression in Session Manager

Overview

3270-type terminals provide fairly well documented features which permit savings in the size of buffers passing through the system between the terminal and the application. Unfortunately, not all applications exploit these features fully, so often more data is sent than is necessary. This means more data flowing on telecommunications links and the more data flowing, the slower the response time at the terminal.

COMPRESS and MISER options

Session Manager provides two methods of 3270 data stream compression which transparently reduce network traffic. These are Runlength compression, and MISER compression. Runlength compression is specified by the COMPRESS parameter; MISER compression by the MISER parameter, on SESSION specifications of the SYSTEM, PROFILE, USER, TERMINAL or APPL statements.

Runlength compression operates by replacing repeated characters in the data stream with the Repeat to Address (RA) order. Any sequence of five or more identical characters is compressed to a four byte RA order. However, efficient applications may use RA orders to build the data stream in the first place so no saving is made. The QUERY STATS command shows the saving made by Runlength compression in messages 434I and 435I.

Some software products provide this function within the application, for example such a system may be implemented in a CICS user exit. However, it is more convenient, and possibly less expensive, to provide this function at one point in the system for all applications. Session Manager provides the ideal place to do this, since the Runlength compression routine is efficient, requires no application changes and is part of the base product, so it can be implemented without incurring any additional cost.

MISER compression is more sophisticated. By 'remembering' what is already on the screen, MISER can remove from an outbound data stream everything which is already displayed and need not be re-written. In addition, MISER can eliminate the Read Buffer command and significantly reduce inbound data streams for some applications. The savings made by these methods are reported in the Q STATS display.

Using both COMPRESS and MISER

The two compression methods do not interfere with each other in Session Manager. They co-operate, and for maximum savings they should both be active. Logically, Runlength compression is first performed on the outbound data stream, then MISER compression. When MISER does have to send data to update the screen it should be a smaller buffer if Runlength compression has occurred first.

Determining compression savings

The QUERY STATS command shows the two-stage compression process, so the percentage savings displayed in the reply for outbound buffers cannot be simply added together. Some Runlength-compressed buffers might not be MISER-compressed, and some MISER-compressed buffers might not be Runlength compressed. Because of this, the two savings are reported separately.

For example, if Runlength compression provides a 25 percent saving (the buffer is compressed to 75 percent of its size before Runlength compression) and MISER compression provides a 40 percent saving (the buffer is compressed to 60 percent of its size before MISER compression), then the final output is 60 percent of 75 percent (= 45 percent) of the original buffer size.

So if the original outbound buffer is 1000 bytes long, and both MISER and Runlength compression are active, Runlength compression giving 25 percent saving and MISER compression giving 40 percent saving, then the compression calculations are:

$$(1000 / 100) * (100 - 25) = 750 \text{ bytes after Runlength compression}$$

$$(750 / 100) * (100 - 40) = 450 \text{ bytes after MISER compression}$$

So the final buffer size = $450 / (1000 / 100) = 45$ percent of its original size, and therefore a 55 percent saving has been made.

The following messages are output in response to a QUERY STATS command:

```
MSG434  Compressed buffers = comp_buf
        Bytes in = bytes_in
        Bytes out = bytes_out
```

Variables

comp_buf is the number of buffers from the application to the terminal which were processed by the Runlength compression routine.

bytes_in is the number of data bytes in the buffers before Runlength compression.

bytes_out is the number of data bytes in the buffers after Runlength compression.

```
MSG435  Percentage saved = perc_saved
```

Variables

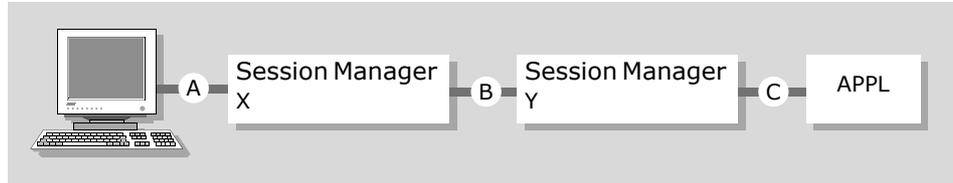
perc_saved is $100 - (bytes_out / (bytes_in / 100))$.

(*bytes_out* and *bytes_in* are as defined above for message MSG434.)

Data compression and remote sessions

Both Runlength compression and MISER compression can be specified when a session is running remotely to reduce the network traffic generated. For example:

Session path using a Session Manager remote session



Session Manager X, is referred to as the local Session Manager, where the user is signed on. Session Manager Y, is referred to as the remote Session Manager, which connects to the application.

Both data stream compression and MISER can reduce the network traffic passing through both path A and path B. The remote Session Manager and the application are, ideally, executing in the same machine. If this is the case, path C is totally within the machine and therefore there is no 'network' component in this path.

Runlength compression

To compress the outbound data streams on both path A and path B, either specify COMPRESS Yes in the local definitions, or specify COMPRESS Yes on the remote APPL definition. Compression will then be processed in the remote Session Manager.

MISER compression

The local session definitions, at Session Manager X, may specify MISER and also RMISER (provided MISER is enabled on this local Session Manager). The MISER specification governs the inbound and outbound data streams from the local Session Manager on path A. The RMISER specification governs the inbound and outbound data streams from Session Manager on path B and path A. Alternatively, in place of the RMISER definition, a MISER definition can be placed on the remote Session Manager APPL definition (provided MISER is enabled on this remote Session Manager).

Recommended MISER definitions for remote sessions

Either

- 1 Specify in local Session Manager (no remote definition required):

```
MISER ERB          /* prevent Session Manager'
                   /*   generated RBUFs.
RMISER ON          /* full MISER processing, that is
                   /*   inbound and outbound data flows
                   /*   and read command emulation.
```

Or

- 2 Specify in local Session Manager:

```
MISER ERB          /* prevent Session Manager'
                   /*   generated RBUFs.
```

and on remote Session Manager APPL statement:

```
MISER ON          /* full MISER processing, that is,
                   /*   inbound and outbound data flows
                   /*   and read command emulation.
```

Summary

The following feature, parameters and variables are required to operate MISER.

Feature, parameter or variable	Description
MISER feature	This must be enabled on the system for MISER to be used.
MISER subparameter	Specifies whether MISER is to be switched on or off, or used selectively. May be specified at session level on the SYSTEM, PROFILE, USER, TERMINAL or APPL statements.
RMISER subparameter	Specifies whether MISER is to be switched on or off, or used selectively, when using remote sessions. May be specified as MISER above, but only in the <i>local</i> Session Manager definitions.
COMPRESS subparameter	Specifies whether Session Manager compression is to be switched on or off, or used selectively. May be specified at session level on the SYSTEM, PROFILE, USER, TERMINAL or APPL statements.
s_miser variable	Indicates the status of MISER for the session.

CHAPTER 13

How to set up and use the CICS front-end

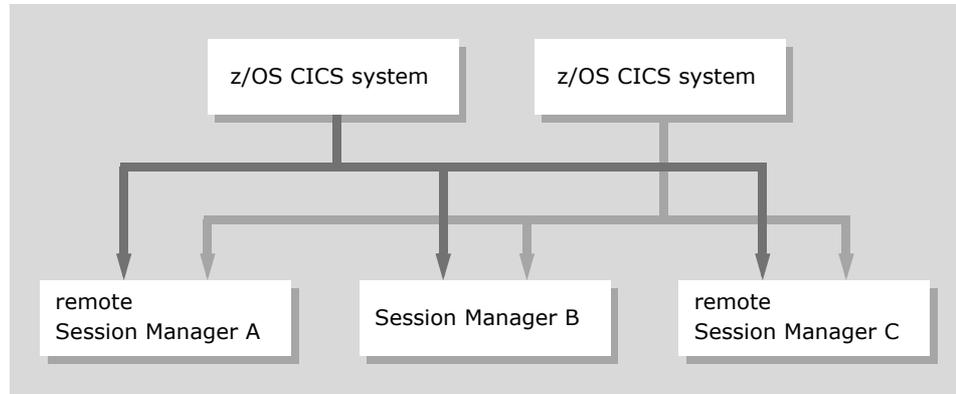
A CICS system can be set up to access a local or remote IBM Session Manager for z/OS system.

The subjects covered in this chapter are:

- 'CICS definitions' on page 339
- 'Accessing Session Manager using the CICS front-end' on page 342
- 'Querying the CICS front-end' on page 345
- 'Terminating the CICS link' on page 346
- 'Console messages from the CICS agent' on page 347
- 'Modifying the supplied starter transaction program' on page 348

Overview

A CICS system can be set up to access a local or remote Session Manager system, as illustrated below.



The CICS agent is merely a vehicle to provide access to one or more Session Manager systems. Once accessed, all activity takes place in the accessed Session Manager.

CICS definitions

Session Manager requires several entries to be made in the Resource Definition File, the CICS CSD. This should be done using the RDO transaction CEDA as normal. Some of the entries are mandatory because Session Manager requires them to operate. Others are Installation dependent.

All the entries described in the sections which follow are defined in group ISZFHAGT. If this name conflicts with Installation naming standards it can be changed.

Program definitions

The programs should be defined to CICS with the following DEFINE statements. (Note that the EXECKEY and DATALOCATION parameters are only for CICS/ESA[®] Version 3.3 and above):

```
DEF PROG(ISZFHAGE) GR(ISZFHAGT) L(A) EXECK(CICS)
                                DATALOCATION(ANY)
```

A starter program is provided for which you need the definition:

```
DEF PROG(ISZFHAXX) L(A) GR(ISZFHAGT)
```

All other operands can be allowed to default.

Notes

- 1 The ISZFHAGE program uses EXEC CICS INQ commands so the group DFHINQUI must be installed.
- 2 The ISZFHAGE program does not access any files so if you have multiple CICS regions it should be installed in the Terminal owning region.
- 3 The supplied starter transaction program is ISZFHAXX but you can replace this with your own and call it whatever you like. You must then define your chosen name and omit defining ISZFHAXX.

Profile definitions

The following DEFINE statements should be entered to define the profiles to CICS:

```
DEF PROF(ISZFHPA) S(ALTERNATE) GR(ISZFHAGT)
DEF PROF(ISZFHPD) S(DEFAULT) GR(ISZFHAGT)
```

All other operands can be allowed to default.

Transaction definitions

The following DEFINE statements should be entered to define the transactions to CICS.

```
DEF TRANS(ISZA) PROG(ISZFHAGE) PROF(ISZFHPA) GR(ISZFHAGT)
DEF TRANS(ISZD) PROG(ISZFHAGE) PROF(ISZFHPD) GR(ISZFHAGT)
```

The default value of NO for DTIMOUT, SPURGE and TPURGE is required for the above transactions.

The transaction names above are fixed and are not entered directly at a terminal. Instead, a 'starter' transaction is used for each target Session Manager system, and this passes such details as the connection to use. The sample 'starter' supplied uses its transaction id as the name of the connection.

The starter transaction definition for Session Manager is:

```
DEF TRANS(ISZT) PROG(ISZFHAXX) PROF(ISZFHPA) GR(ISZFHIBM)
```

All other operands can be allowed to default. The transaction id does not need to be ISZT. In fact, if you have multiple Session Manager systems you will need to define multiple starter transactions, each with a different transaction id. The transaction id is used in the connection and session definitions. The program name must be the one you have chosen for the starter transaction program.

The example shows the definition in a different group to the main definitions. This is not a requirement. If Session Manager is the only product using the CICS agent, then the transaction can be defined in the same group.

Connection and session definitions

Assuming that Session Manager is using the applid ISZSMGR, and that the CICS starter transaction name is ISZT, the following definitions are required:

```
DEF CONN(ISZT) GR(ISZFHIBM) NETNAME(ISZSMGR) AUTOCONNECT(YES)

DEF SESS(ISZS) GR(ISZFHIBM) CONN(ISZT) AUTOCONNECT(YES)
    MODENAME(#INTER) MAX(2,2) SENDSIZE(4096)
    RECEIVESIZE(4096)
```

Notes

- 1** The modename #INTER is supplied with the VTAM default logmode table. Alternatively, you can use any LU6.2 modename, providing it is defined in the default logmode table, or defined in a logmode table referenced by both MODETAB entries on the VTAM APPL definitions for Session Manager and CICS.
- 2** If access to more than one instance of Session Manager is needed, the above connection and session definitions can be duplicated as required, using different connection and netname values. The starter transaction id must match the connection value, that is, in the above connection and session definitions, where ISZT appears change this to match the associated transaction id.
- 3** This example shows the connection and session definitions in a different group to the main definitions. This is not a requirement. If this is the only product using the CICS agent, then they can all be defined in the same group.
- 4** ISZFHAGE is designed to multiplex the data flow between CICS and Session Manager across two conversations. This means that the MAX value does not control the number of simultaneous users. The extra two conversations from a single CICS region are unlikely to invalidate the EAS value in your VTAM definitions, but if multiple CICS regions are planned it should be checked.
- 5** ISZFHAGE uses the 'pseudo-conversational' design. In addition to the short-lived tasks which interact with the terminals, it needs two long running non-terminal tasks per active connection definition and one other long-running service task.

For example: A CICS region connected to two Session Manager systems will have five long-running tasks (two plus two plus one) with extra short-lived tasks being created to send and receive data from the terminals.

Installing the resource groups

The groups for the preceding entries must be defined using the CEDA transaction as follows:

```
INSTALL GROUP(ISZFHAGT)
INSTALL GROUP(ISZFHIBM)
```

Accessing Session Manager using the CICS front-end

Using the Session Manager definitions shown in this chapter, and assuming that the LU ISZSMGR has been opened by a Session Manager system running with jobname PRODISZA, you can signon to Session Manager by entering at a CICS terminal your defined transaction id, in this example, ISZT.

If the CICS region is already linked to the PRODISZA job, the Session Manager signon screen should appear.

Startup messages

If this CICS region is not already in contact you may see, depending on timing, one or more of the following messages:

```
Logon pending, contacting ISZT
```

or once the primary conversation has started:

```
Logon to ISZT in progress
```

or after receiving an initial reply from Session Manager:

```
Logon to Session Manager v.rmm PRODISZA in progress
```

The above should be followed by messages describing the status of the conversations:

```
Primary conversation active
```

is normal, but occasionally you might see the intermediate states:

```
Primary conversation allocated  
Primary conversation being started
```

Once the secondary conversation is active the next display should be from Session Manager, but you might see:

```
Secondary conversation being started  
Secondary conversation allocated  
Secondary conversation active
```

Other messages

Some of these messages indicate problems:

```
Connection ISZT not defined
```

Check that the CEDA DEFINE has been specified and that the group has been installed.

Connection ISZT is not LU 6.2

APPC is the default protocol for access method VTAM so this should not occur. Issue a CEDA VIEW CONN command in CICS to get more information.

Connection ISZT Out of Service
Connection ISZT is going Out of Service

Issue a CEMT INQ CONN in CICS to confirm the status. If you have the requisite authority, try bringing the connection into service.

Connection ISZT released

Inservice connections can be 'Acquired' or 'Released', if you have the authority, issue CEMT SET CONN(XXXX) ACQUIRE.

Connection ISZT being acquired

CICS is attempting to acquire connections. The CICS agent retries every few seconds, or more often if you press Enter.

Primary conversation failed
Secondary conversation failed

Check that you have the correct netname in the DEF CONN definition and if so check the Session Manager log for error messages.

Primary conversation rejected (SYSID already known)

Session Manager uses the CICS SYSID to identify a CICS region so it cannot process requests from two regions having the same SYSID. This message may also occur if the structures created to handle a previous connection from this region have not been deleted. Check the Session Manager log for error messages.

Secondary conversation rejected (mismatch)

Once the primary conversation has started, Session Manager sends CICS a token which must be passed as part of the Secondary conversation parameters. This message indicates an unexpected value.

Press ENTER to continue, PF3 or CLEAR to exit

When the CICS agent issues messages it may prompt on the last line of the screen with this message. This indicates that it is still trying, or will retry in a few seconds. ENTER causes an immediate retry, PF3 or CLEAR will abandon your signon.

Session Manager v.rmm PRODISZA session terminated

When you sign off from Session Manager you may receive a message similar to this. However, the starter transaction can be configured to run either a transaction or program at exit and if either of these options are used the terminated message will not appear.

Press ENTER to continue

When the CICS agent has messages displayed it will prompt on the last line and wait for the ENTER key being pressed before running the selected transaction or program.

Querying the CICS front-end

Issue the Session Manager command:

```
ZQuery CICSlink name
```

Where *name* can be specified using wildcard characters and defaults to asterisk (*). For each CICS region that Session Manager is linked to, this command displays message ISZ0197I showing each CICS region's SYSID, the name and level of the CICS system, and the jobname. For example:

```
ISZ0197I CICSLINK SYS1 CICS/ESA 4.1 PRDCICS
```

Note To list the active CICSLINKs you must use ZQ CICSlink, not ZQ TASK CICSLINK. To list the active terminals for a system use ZQ TASK C-SYS1 where SYS1 is the CICS region sysid.

Terminating the CICS link

In normal circumstances the link will terminate on its own, or at Session Manager shutdown. This command is only required when a problem is encountered with the link, then issue the Session Manager command:

```
STOPCICSlink name
```

Where *name* is required and cannot use wildcard characters, it must be specified exactly.

The link to the named CICS region will be terminated, and all users signed on from that region will be logged off from Session Manager.

Console messages from the CICS agent

The CICS agent issues various messages to report activity during startup and termination.

```
ISZ0030I CICSLINK MANAGER active
ISZ0030I LU62RMA ISZSMGR active
```

These messages are issued during startup. ISZSMGR is the Terminal ACBname.

```
ISZ0030I LU62CNOS DBDCCICS active
ISZ0076I CNOS DBDCCICS modename - RCVD 002,002,000
ISZ0076I CNOS DBDCCICS modename - SENT 002,002,000
ISZ0031I LU62CNOS DBDCCICS terminated
```

When a CICS region first contacts Session Manager these messages should appear. DBDCCICS is the CICS region's ACBname. *modename* is the modename of the CEDA DEFINE SESSION, defaulting to #INTER. These should be followed by:

```
ISZ0030I CICSLINK SYS1 active
ISZ0030I C-SYS1 17A0 active
```

SYS1 is the CICS region's SYSID and 17A0 is the Terminal ID within that region. If more than one terminal signs on extra messages appear:

```
ISZ0030I C-SYS1 17A1 active
```

where 17A1 is another terminal ID within region SYS1.

```
ISZ0031I C-SYS1 17A0 terminated
```

Appears when a terminal signs off.

```
ISZ0031I CICSLINK SYS1 terminated
```

Appears when all the terminals from a region have signed off. The next terminal to signon from SYS1 will start the process again from:

```
ISZ0030I CICSLINK SYS1 active
```

Modifying the supplied starter transaction program

The sample starter transaction program is ISZFHAXX (in library .SISZCONF). The supplied code in ISZFHAXX can be modified, or completely replaced, and use any suitable language – it need not be Assembler and the program name need not be ISZFHAXX, although your chosen name will need to be specified in the CICS program definitions. Sample assembly and linkedit JCL is provided in ISZCCJCL (in library .SISZCONF).

To start a session, the program must issue the following:

```
EXEC CICS START TRANSID('ISZD') TERMID(EIBTRMID)
FROM(PARAMS) LENGTH(P_TOTL)
EXEC CICS RETURN
```

where PARAMS has the following structure:

```
PARAMS      DS  0H
P_TOTL      DC  AL2(TLEN)  Total length of structure
P_HDRL      DC  AL2(HLEN)  Length of header section

              DC  CL4'OPEN'  Required parameter

P_SYSID     DC  CL4' '      Name on DEF CONN eg. ISZT
```

* The sample code sets P_SYSID to the transaction ID.

```
P_XCTLP     DC  CL8' '      Program to XCTL to at exit or blanks.
P_XTRAN     DC  CL4' '      Transaction to start at exit or blanks.
```

* You should only set one of the above non-blank.

```
P_USR0      DC  AL2(U-PARAMS)  Offset to user data
P_USRL      DC  AL2(UEND-U)    length of user data
```

* If user data is not required set P_USR0 and P_USRL to zero
 * otherwise the value at P_USR0 must not be less than that at
 * P_HDRL and P_TOTL must be at least P_USRL+P_USR0.

* Future versions of ISZFHAGE may support extra values here. Any
 * padding bytes must be zero.

```
HLEN        EQU  -PARAMS
```

* Anything here will be ignored

```
U          DS   nnnC
```

```
User data max 256 bytes
```

```
UEND      DS   0C
```

```
* Anything here will be ignored
```

```
TLEN      EQU *-PARAMS
```

The supplied sample code issues an EXEC CICS RECEIVE and then extracts the data after the transaction name. This is trimmed of leading and trailing blanks before being moved into the above structure. Within Session Manager this value is available in the session variable *t_appcdata*.

CHAPTER 14

Session Manager and TCP/IP

Transmission Control Protocol/Internet Protocol (TCP/IP) is a network architecture which enables communication between different types of computer and different physical networks. This chapter introduces you to the components of TCP/IP and discusses support for TCP/IP in Session Manager.

The subjects covered in this chapter are:

- 'Introduction to TCP/IP' on page 352
- 'Support for TCP/IP in Session Manager' on page 355

Introduction to TCP/IP

TCP/IP links different types of computers and different physical networks logically into a single, large virtual network, an 'internet', and provides infinite possibilities for the distribution of systems and data and for co-operative processing.

TCP/IP consists of a set of standard protocols, or rules, for networking communications, which specify in detail how information is sent across the network and how various services are implemented. The protocols, therefore, represent the fundamental basis for the way parts of an internet interconnect and operate with each other.

TCP/IP takes its name from the two most important elements of this set of protocols:

- **IP (Internet Protocol)**
which provides a standard interface to any type of network. IP is the basic transport mechanism for routing packets of data and handles routing addresses; however, it does not ensure that data has been received.
- **TCP (Transmission Control Protocol)**
which is concerned with the reliable delivery of the data. TCP breaks up the data into packets for sending via IP and reassembles them at their destination; it also retransmits data where necessary and reports any errors in transmission.

Remote nodes

Each TCP/IP host is known by an internet address. The internet address of the remote host to which data is sent becomes the destination address for routing, and accompanies the data in its passage over the network(s).

A host may have more than one address. Typically, a host could have an IPv4 address (as is usual in the current standard) and a second address in the newer IPv6 protocol. Session Manager supports both IPv4 and IPv6 internet addresses.

An IPv4 internet address is a 32-bit address expressed as a series of four decimal numbers separated by dots (periods), each number being the value of one byte. An IPv6 address is 128 bits long and is expressed as the hexadecimal values of 8 consecutive halfwords separated by colons. The IPv6 protocol was introduced to cope with the impending exhaustion of IPv4 addresses.

Examples of internet addresses are:

IPv4 address	Expressed as
0A01285A	10.1.40.90
IPv6 address	Expressed as
200109800FFE00010000000000000042	2001:980:ffe:1:0:0:0:42

Note that both formats allow for the suppression of leading zeroes in each division. In addition, IPv6 addresses allow for the omission of consecutive halfwords of zero by specifying two colons to signify the start and end of these divisions.

For example, the above address,

```
2001:980:ffe:1:0:0:0:42
```

can be expressed as

```
2001:980:FFE:1::42
```

The numeric format of internet addresses makes them difficult to use. It is therefore preferable to refer to them by symbolic names. Before data is sent to a host identified by a symbolic name, this name is converted to an internet address by a name server.

Note Only the symbolic name is supported in Session Manager configuration statements for IPv6 addresses.

A symbolic name consists of labels separated by dots, each label having a maximum of 63 characters and the whole name having a maximum of 255 characters. A complete name which can be translated to an internet address is known as a fully qualified domain name. This is hierarchical name wherein each label from left to right represents an increasingly higher domain level within an internet. As with an internet address, the fully qualified domain name can be divided into a network name and a host name.

An example of a fully qualified domain name is:

```
zosa.ibm.com
```

where 'zosa' is the host name and 'ibm.com' the network name.

Hosts in the local domain can be referred to by the host name only; the local resolver combines the host name with the domain name before sending a request for the internet address to the name server.

The name of the host to which Session Manager is to send data must be specified for each TCP/IP session. Unless the name of the host is always specified as an internet address, one or more DD statements must be present in the Session Manager JCL. For more information, see 'The TELNET application' on page 354.

Each application must be identified by its TCP/IP name, which is the name by which it is known to the remote system; this name is case sensitive and must therefore be specified in exactly the same way as the remote host uses it.

Ports

Each application has its own port for communicating with other applications. A port is identified by a number between 0 and 32767. TCP allocates a port number to Session Manager for sending data and this is known as the source port. Communication between applications is by connection between sockets, each socket having an address which is a combination of the host's internet address and the application's port number.

The display of the port number in messages containing resolved internet addresses will vary depending on whether the address is IPv4 protocol or IPv6 protocol. Taking the addresses used in the examples above, if the port assigned is 23 (the default Telnet port), for the IPv4 address it would appear as follows:

```
10.1.40.90:23
```

This convention would cause confusion with an IPv6 address, as the colon is used as the division separator in this protocol. Therefore a port number of 23 with the IPv6 address above is displayed as:

```
2001:980:FFE:1::42 :23
```

TCP/IP software

Session Manager supports IBM's TCP/IP product.

The TELNET application

The TCP/IP TELNET application provides access to other remote hosts across an Internet network. During logon the details of the actual terminal characteristics can be negotiated. This negotiation uses commands to establish what is allowed – for example, full screen functions and what transmission protocols are to be used.

The Eclipse plug-in interface

The Eclipse plug-in interface communicates with Session Manager via TCP/IP.

For more information, see 'Enabling Eclipse support' on page 126 and the *User and Administrator* manual.

Support for TCP/IP in Session Manager

Overview

Support for TCP/IP in Session Manager uses the TELNET application: server support using the TN3270 protocol; client support using the TN3270, VT100/VT200, ANSI or Network Virtual Terminal (NVT) protocols of TELNET. Support in the Session Manager Configuration is supplied by the following SYSTEM statement parameters:

```
TCP [ Yes | No | ON | OFF ]
    [ DISPLAY [ Yes | No | ON | OFF ] ]
    [ STN3270 port_no [TN3270E [ Yes | No | ON | OFF ] ]
      [TN3270E_CONNECT [ Yes | No | ON | OFF ] ] ]
    [ [ IBM ] [ IUCVname resource_name ]
      [ [ TRACE [ Yes | No | ON | OFF ] ]
        ECLIPSEServer portnumber [DIAGS Y|N]
```

For more information on TN3270E support, see 'Additional support for TN3270E' on page 355; for full details of all the above parameters, see the 'SYSTEM statement' chapter in the *Technical Reference*.

Note For prerequisites, see 'Prerequisites' on page 61.

TELNET/TN3270 server support

Any TN3270 client can access Session Manager as a server using the TN3270 TELNET server support within the TCP/IP stack (which creates a virtual LU in VTAM for Session Manager). For users of Session Manager running IBM's TCP/IP interface, Session Manager TN3270 Server Support provides an alternative method which can be more efficient and also enables Session Manager to operate without VTAM.

With Session Manager TN3270 Server Support, TCP/IP users running TN3270 can connect directly to a Session Manager Signon or Menu panel. Session Manager then acts as a TCP/IP TELNET Server.

Support for this facility is supplied by the following STN3270 parameter of the SYSTEM TCP parameter (see above):

```
STN3270 port_no
```

Start a TN3270 server at port number *port_no*.

Additional support for TN3270E

If 'TCP STN3270' is specified then the TN3270E and TN3270E_CONNECT parameters can optionally be specified:

```
TN3270E [Yes|No|ON|OFF]
```

The default is TN3270E NO.

If you specify TN3270E Y then the Session Manager TELNET server will negotiate with TN3270 clients to use certain TN3270E protocols. Forward sessions subsequently initiated will, by default, use LU2 protocols (as opposed to LU0 protocols). In addition, the client can pass a device name which can optionally be used by Session Manager when allocating ACBs for forward sessions (see TN3270E_CONNECT below).

TN3270E_CONNECT [Yes|No|ON|OFF]

The default is TN3270E_CONNECT NO.

If you specify TN3270E_CONNECT Y, which implies TN3270E Y, then TN3270E support is taken a step further. When TN3270E_CONNECT Y is in effect, the device name supplied by the client will be used to nominate a virtual terminal (VTAM ACB) which may be used when subsequently establishing forward sessions. This device name may relate to either:

- **A specific ACB**

Care should be exercised if using a specific ACB as the device name. In this case, there is the possibility of multiple Session Manager userids being allocated the same virtual terminal name, which would lead to potential conflicts in applications using the ACB (see 'Examples' on page 357).

If during terminal connection Session Manager is unable to open the specified ACB then the connection will fail.

- **A range of ACBs**

In order for the value to be related to a range of ACBs, it should refer to the name of a special Session Manager APPL definition for which there is an associated RANGE definition. (If no matching APPL definition is found then the value will be assumed to be a specific ACB name.)

If during terminal connection Session Manager is unable to open the first available ACB in a range then it will attempt to select another ACB in the specified range. If an ACB in the specified range is not available then the connection will fail.

Variables set when the user signs on

These variables are set when the user signs on:

Variable	Description
t_user_acb	This user-modifiable variable is set to the name of the virtual terminal selected.
t_user_appl	If the name of the virtual terminal was selected from a range then this read-only variable is set to the name of the APPL definition used to determine the range.

Reconnecting and T_TN3270E_CONNECT Y

When T_TN3270E_CONNECT Y is in effect, TN3270E Y processing is invoked automatically during connect processing. If this processing caused an ACB to be selected from a range then:

- Ownership of that ACB will initially be allocated to the TN3270E device.
- If the user logs on to a Session Manager userid then ownership of that ACB will be transferred to that userid. Alternatively, if the user reconnects to a Session Manager userid then Session Manager will deallocate the ACB just allocated to the TN3270E device, freeing it up for other users.

Note If the userid previously had a virtual terminal associated with it then the user will continue to use that same ACB; if not then, as before, there will be no virtual terminal associated with the user.

Examples

If the device name is set to ACBUSERA, which is the name of a specific VTAM ACB, then this sequence of events might take place:

- TN3270E user connects. Session Manager signon screen is displayed.
- TN3270E user signs on as USERA. Variable `t_user_acb` is set to ACBUSERA.
- User USERA starts a session with CICS using ACBUSERA.
- User USERA disconnects from Session Manager leaving the CICS session still active, and returns to the signon screen.
- TN3270E user signs on as USERB. Variable `t_user_acb` is set to ACBUSERA.
- User USERB attempts to start a session with CICS using ACBUSERA.
- CICS rejects session request because ACBUSERA is already in use.

Using the 'range of ACBs' method (that is, setting the device name to the name of an APPL definition) avoids this problem. For example, if the device name is set to the name of an APPL definition associated with a range of ACBs from ACB001 to ACB099 then this sequence of events might take place:

- TN3270E user connects. Session Manager allocates first free ACB in range, ACB001, to TN3270E device. Session Manager signon screen is displayed.
- TN3270E user signs on as USERA. Session Manager transfers ownership of ACB001 to user USERA. Variable `t_user_acb` is set to ACB001.
- User USERA starts a session with CICS using ACB001.
- User USERA disconnects from Session Manager leaving the CICS session still active, and returns to the signon screen. Session Manager allocates the first free ACB in the range (now ACB002, as ACB001 is still allocated to user USERA) to TN3270E device.
- TN3270E user signs on as USERB. Session Manager transfers ownership of ACB002 to user USERB. Variable `t_user_acb` is set to ACB002.
- User USERB starts a session with CICS using ACB002.

Read-only variables associated with TN3270E support

These read-only variables are associated with TN3270E support:

Variable	Description
<code>t_tn3270e</code>	If Session Manager has successfully negotiated with the TN3270 client to use TN3270E protocols then this variable returns a value of 'Y'. Otherwise, and for all other terminal types, this variable returns a value of 'N'.
<code>t_tn3270e_name</code>	Returns the device name, if any, supplied by the TN3270E client at connection time.

TELNET/TN3270 client support This provides access to remote applications running on any host connected over a TCP/IP network. It allows a Session Manager user to have one or more TCP/IP TELNET Client connections. These appear as conventional Session Manager sessions that are configurable in the normal way and which co-exist with any other conventional sessions.

Network Virtual Terminal (NVT) is supported by most TELNET Servers and is a line mode interface. If the Server can support the full screen TN3270 connection then this is used by default in preference to the line mode NVT connection. NVT provides simple 'line-by-line' terminal support using the TELNET NVT (Network Virtual Terminal) protocol. The TN3270 session appears identical to a conventional Session Manager session.

This facility allows a Session Manager user to access and run any Server application which is available to the TCP/IP network. The host on which that application runs must have a TELNET Server supporting NVT. The Session Manager user interacts with the remote application using the 3270 terminal (or PC emulated 3270) in line-by-line mode.

Session Manager can start a session with any TN3270 Server using a TSO address space which runs a TCP/IP client. With Session Manager TN3270 client support however, the need for a dedicated TSO address space is bypassed, resulting in performance improvements, particularly when a user has, for example, three TN3270 sessions.

TELNET/VT220 client support VT220/VT100 emulation is supported. This enables users to access a UNIX application from a 3270 terminal.

TN3270E client support From Session Manager version 1.3.05 TN3270E emulation is supported. This enables users to access TN3270E Server applications.

Starting and stopping TCP/IP

You can stop and restart the Session Manager TCP/IP manager (and therefore the TN3270 Server and any TELNET client sessions) using:

- **STOPTcp**
This command stops the Session Manager TCP/IP manager. It has a default authorization level of 9.
- **STARTcp**
This command starts the Session Manager TCP/IP manager. It has a default authorization level of 9.

Configuring TCP/IP client sessions in Session Manager

The TELNET client session operates and is configured in the same way as other Session Manager sessions.

- 1 To ensure Session Manager recognizes the session as a TELNET client session, specify the APPLID parameter as TCP_1.
- 2 Specify the target TELNET Server's URL (Uniform Resource Locator) in the DATA parameter in the following format:

```
TELNET://[[userid][:password]@]host_IP_name|addr[:port_no][/]
```

or

```
TN3270://[[userid][:password]@]host_IP_name|addr[:port_no][/]
```

or

```
TN3270E://[[userid][:password]@]host_IP_name|addr[:port_no][/]
```

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

The *userid* and *password* are 'advisory only' since all TELNET clients, including Session Manager, cannot enter these fields automatically into the session. The values are contained in the session variables `s_telnet_user` and `s_telnet_pswd`. These could be entered on TELNET Server signon screens via session scripts.

The *host_IP_name* or *addr* can be accessed via the variable `s_telnet_host`.

A port is an address in the range 0-32767. Each application has an individual port address. Standard applications, for example FTP, have pre-assigned port numbers and are known as 'well-known' ports. Ports for use by applications are assigned by the Portmap function of TCP/IP.

If the *port_no* is not specified, the value in variable `ln_telnet_port` is used. If this value is not specified, port 23 is used. The port number in use by the session can be accessed by the variable `s_telnet_port`.

TELNET client session modes

The Session Manager TELNET client will operate in one of several modes:

- 1 TN3270
TELNET full-screen 3270 session.
- 2 VTxxx
Emulating ANSI or VT100 or VT220 protocols.
- 3 NVT
TELNET default line mode.
- 4 TN3270E
Enhanced TELNET.

If the Session Manager session logmode is not specified, Session Manager will negotiate with the server to establish the best TELNET protocol to use. Session Manager will initially try TN3270 protocols. If the server does not support these, Session Manager will try VTxxx protocols. If the server does not support these, it will operate in NVT mode.

If the Session Manager session logmode is specified, it must be one of the following:

Dxx327nm	where:
	<ul style="list-style-type: none"> ▪ x is any character. ▪ n is either 8 or 9 (8 does not use 3270 Query, 9 does use 3270 Query). ▪ m is the 3270 model, either 2, 3, 4 or 5.
S3270	IBM-3277-2 terminal type. S3270 is used if no logmode is specified and the user's terminal has no alternate screen size.
VT220	VT220 terminal type
VT100	VT100 terminal type
ANSI	for ANSI mode
LINEMODE	for NVT mode

Note The Server can, however, override the specified logmode. For example, it may only support model 2 screens and so the session will operate in model 2 in spite of a model 5 logmode or terminal model.

- **If the TN3270 protocols have been negotiated**
the session operates and appears identical to a standard Session Manager session.
- **If VTxxx or ANSI protocols are to be used**
(either because a logmode has been specified or due to negotiation with the server), then this will operate as described in 'VT220/VT100 emulation' below.
- **If line-by-line mode is to operate**
the following applies:

The TELNET panel is displayed. This has the standard Session Manager format of a header, contents, and trailer lines. The Contents lines are supplied using the subscriptable `t_data` variable. As each output line is generated, it is placed in the lowest line of the content. Previously generated output lines are then moved up a line. The supplied Session Manager panel TELNET can be found on the media in ISZPANS (in library .SISZCONF). You can specify an alternative panel using the `lc_telnet_panl` variable.

The trailer should contain the `t_command` variable.

Variables that can be used in this panel are described in the *Panels, Scripts and Variables* manual in the 'Session Manager variables' chapter. See also 'TELNET PF keys and client commands' on page 367 for details of local variables that can be used in this panel.

VT220/VT100 emulation

VT220/VT100 Emulation extends the TELNET client support within Session Manager. It enables a 3270 terminal to communicate with a VT220, VT100 or ANSI (UNIX) application using the TELNET protocol.

The emulation is provided in supplied Scripts and Panels. VTxxx PF keys are emulated using specific cursor position, or by escape sequences. PF keys and escape sequences can be redefined by an Installation if there are special requirements.

Using a VTxxx session on a 3270

VTxxx 24x80 and 24x132 sessions can be displayed on all 3270 models 2, 3, 4 and 5. In some cases this will leave an area around the VTxxx display which can be used for information display and for command areas (for Session Manager or TELNET). In other cases, there will be no such spare area and the VTxxx display will need to be zoomed or focused in order to see the relevant part.

Special commands for VTxxx mode

These special commands are available for VTxxx mode:

CURSOR UP | DOWN | LEFT | RIGHT

This command is interpreted according to the current setting of the Cursor key application switch and the result treated as a DATA command. The cursor positioning command is sent after any input data has been sent.

DATA *keystroke-names*

If the cursor is in the VTxxx input area, the keystrokes are sent after the screen data has been sent (as for SINGLE command).

FOCUS [TOP | BOTTOM | LEFT | RIGHT | TOP LEFT | TOP RIGHT | BOTTOM LEFT | BOTTOM RIGHT]

When a VTxxx screen cannot be fully displayed on a 3270 terminal, Session Manager attempts to display the part that is of interest. This is assumed to be the area where the cursor is positioned. If this is not the area required, then the FOCUS command can be issued to set the VTxxx display at the required position.

If setting the focus causes the cursor to be hidden (because it is on part of the screen not currently displayed), issuing FOCUS without an operand sets the screen to the area where the cursor is.

If the VTxxx screen is 24x80 then the ZOOM command can be used instead of the FOCUS command to display the VTxxx screen in full-screen mode.

HIDE

The input area is changed to non-display so that a password can be entered. The HIDE command is usually assigned to PF21.

HOLD

When a lot of data is transmitted to the 3270 terminal, the display will scroll. Issuing the HOLD command, or pressing the PF key assigned to HOLD, will stop the display from scrolling. Scrolling is resumed when the HOLD command is issued again. When the display is in 'hold' state, the held data is kept in storage so that it can be displayed when scrolling is resumed. This could, however, use a lot of storage.

QUIT

Unconditionally terminates a session.

SETUP

Displays the Setup panel. Setup state allows options relating to the emulated terminal to be viewed and changed.

This can be done at any time.

SINGLE

Usually assigned to a PF key. This sends data to the application without any control codes at the end of the datastream.

ZOOM

When a 24x80 VT220 screen is displayed on a 3270 model 2 there is no spare area for error messages or a command line. The ZOOM command can switch between displaying the VT220 screen in full and 'panel mode'. In panel mode, the whole VT220 screen will not be displayed because there will be some header and trailer lines showing on the 3270 terminal to provide a command line and message area. In this mode the VT220 screen can be scrolled.

Ideally, the ZOOM command should be assigned to a key, usually PA1.

Sending data to the application

Data can be sent to the application by pressing a key assigned to the RETURN command (usually the Enter key), or the SINGLE command (usually PF6). SINGLE is similar to RETURN but does not add any control codes to the end of the datastream, it just sends the data. SINGLE can be used when an input field provided by the emulator is not long enough to contain all the data to be sent. RETURN sends the data and then emulates the VTxxx Return key.

The data between the VTxxx cursor position and the IBM cursor position is transmitted to the application when the Enter key or a function key is pressed. This means that if a typing error is noticed and corrected before pressing the key, the cursor must be replaced at the end of the data to be transmitted.

The VTxxx cursor position is where the 3270 cursor is positioned after a write to the screen. It is not visible as a character so the position must be remembered when entering and sending data. Be aware that because the emulator needs to provide a field in which a value can be typed, the last character sent by the application is changed by the emulator to a Start Field command. This means that the last character sent will not appear on the 3270 screen. If input is required, start typing where the 3270 cursor is.

Note If the input field is just 'e', 'E', 'f' or 'F' it may be treated as an editing or function key, see 'Function and editing keys' on page 363. To transmit a single character field holding one of these characters as data, follow the desired character with the TELNET escape character displayed on the panel.

Control codes and escape sequences

Escape sequences can be used to emulate control codes or special characters which are not on the keyboard. See also 'Escape characters' on page 366.

Some applications use Control characters generated by holding the Ctrl key and pressing another key. The TELNET client defines an escape character (normally X'4A' which is a cent sign (¢) on a US keyboard, dollar (\$) on a UK keyboard). This can be used before an alphabetic character to generate the control code. For example, if the application asked for 'ctrl-A' you would type '\$A'. This escape character is displayed as part of the frame when not ZOOMed.

If a control sequence is used to end the input field, it will cause a RETURN command to be processed as SINGLE. Similarly, the TELNET escape character used as the last character of the input field will also cause a RETURN command to be processed as SINGLE.

The VTxxx escape is hexadecimal code X'1B'. This must not be confused with the TELNET client defined escape character described above. To send X'1B' after the data from the input field a PF key (default PF11) can be defined with a DATA command. This is used in a similar way to that described in 'Arrow keys, Backspace, Linefeed and Tab' below.

The X'1B' code is commonly used with a single character code, for example ESC A often represents 'Up'. To support this the TELNET client has a command, ESCAPE (normally PF5), that sends X'1B' followed by the first character of the input data.

Function and editing keys

Function keys

All the VTxxx family have keys marked PF1 to PF4. In addition the VT220 supports keys for F6 to F20, although some of these have other names on the keytops, for example, F15 is marked 'Help' and F16 is 'Do'.

PC emulation programs may also support F1 to F5.

To send a function key sequence to the application enter a single 'F' or 'f' in the input field and press the corresponding 3270 PF key. Alternatively, place the cursor on the F-key string highlighted in the top left hand corner of the screen and press the required F key.

By default, 3270 PF1–PF4 map to VTxxx PF1–PF4, and PF21–24 generate VTxxx F1–F4. This can be swapped around using Setup.

Editing keys

The VT220 has 'editing' keys. If the application requires these they can be obtained by entering a single 'E' or 'e' and pressing PF1 to PF6:

PF1 => 'Find', F2 => 'Insert Here', F3 => 'Remove',
PF4 => 'Select', F5 => 'Prev Screen', F6 => 'Next Screen'

These conventions take precedence over the normal usage of function keys to replace Arrow keys, Tab and so on. To use a single 'e', 'E', 'f' or 'F' with the normal PF key definitions, follow the desired character with the TELNET escape character.

Arrow keys, Backspace, Linefeed and Tab

Some VTxxx applications use cursor positioning keys    , Backspace, Linefeed and Tab.

The 3270 arrow keys are processed by the 3270 itself and cannot be sent to the application. Instead PF keys assigned to the CURSOR command replace the VTxxx arrow keys. The default setup is designed to make the shape of the 'PF' arrow keys match the shape of the VTxxx arrow keys when used on a 3270 that has two rows of PFkeys, that is PF13 to PF24 being above PF1 to PF12. That is:

PF20 maps to 

PF7 PF8 PF9 map to   

Backspace, Linefeed and TAB are also replaced by PF keys, normally PF12 to PF14, which have been assigned the DATA command with an appropriate operand.

The DATA command (and also the CURSOR command) sends any relevant data from the input field *before* sending the special sequence.

If the input field data is just 'e', 'E', 'f' or 'F' special rules apply, see 'Function and editing keys' earlier.

Using the VT220 Setup panel

The Setup panel is invoked by the `SETUP` command being entered in the Session Manager command line as shown in the following unzoomed VT220 sample screen:

```

F-key -----
20 NOV 02          VICTORIA UNIVERSITY OF WELLINGTON          11:09pm
                   Public Access Catalog

      You can search by any of the methods listed here.
      Type in the number of the search you want.

1.  Title browse           10. Closed Reserve & 3-Day Loan
2.  Title keyword         11. Contents keyword
3.  Personal name browse  12. Index to Legal Periodicals
4.  Corporate name browse 13. Library Literature (index)
5.  Corporate name keyword 14. Architecture Lib. New Titles
6.  Subject browse        15. Review Personal Record
7.  Subject keyword       16. Library hours & notices
8.  Series browse         17. Logoff
9.  Series keyword

Enter your selection(s) and press <enter> :
S=Shortcut on, BB=Bulletin Board, ?=Help

-----
VT220   Esc $          ==>  setup
1:Help   2:Telnet    3:Quit    4:ISZcmd  5:Escape  6:Single
7/8:Up/Down 9/10:Left/Right 11:Data Esc 12:Backspace 13:Linefeed 14:Tab
21:Data Del 22:Telnet IP 23:Hide   24:Hold   PA1:Zoom

```

Pressing Enter will then display the Setup panel:

```

Session Manager          VT220 Setup
User user dd/mm/yyyy hh:mm:ss
                        LU luname
Keyset = GC_VTXXX      -> British
                        Flemish
Minimum key help lines = 3
                        French Canadian
                        Danish
                        Finnish
                        German
Clear Display          Dutch
Reset Terminal        Italian
80 Columns            Swiss (French)
Auto Wrap              Swiss (German)
No New Line            Swedish
Numeric Keypad         Norwegian
Application Cursor Keys
Exit                   French/Belgian
                        Spanish

Tab  ....+....1....+....2....+....3....+....4....+....5....+....6....+....7
      ....+....8....+....9....+....10....+....11....+....12....+....13..

Command ==> _____

```

The fields on this screen are tabbable using the 3270 Tab keys or the arrow keys. Any value changed will be set for the current session only.

Note The **Clear Display** field and the fields below it can be selected by cursor positioning. The values of some of these can be toggled.

Field name	Description
Keyset	This field can be overtyped. It can take one of the following values: GC_VTXXX, GC_CSFI_0 or GC_CSFI_1. The first is for VT220, the last two are for VT100 and VT101 respectively and have different key settings. The values set for the keys can be found in the supplied member ISZSTEL (in library .SISZCONF).
Minimum key help lines	Can take a value from 1 to 9.
Clear Display	Clears the screen when you exit setup.
Reset Terminal	Resets a number of terminal operating features to a default setting used by most application programs.
80 Columns	Toggles between 80 columns and 132 columns.
Auto Wrap	Toggles between Auto Wrap and No Auto Wrap. With Auto Wrap, characters received after the right margin appear in the first character positions of the next line. With No Auto Wrap, the characters overwrite the last characters on the current line.
No New Line	Toggles between No New Line and New Line. With No New Line, the Return key sends a carriage return only. With New Line, the Return key sends a carriage return and a line feed.
Numeric Keypad	Toggles between Numeric Keypad and Application Keypad mode. When a real VTxxx is in Application mode, the keys on the auxiliary keypad send control sequences. However, these cannot be directly emulated by keys on a 3270 device. Instead, the control sequences can be generated using the DATA command with the following operands: AUX-0 to AUX-9 AUX-ENTER AUX-COMMA AUX-DASH or AUX-MINUS AUX-PERIOD or AUX-DOT An indication of Aux is displayed in the bottom area of an unzoomed frame as a reminder that the application expects control sequences.
Application Cursor Keys	Toggles between Application Cursor Keys and Normal Cursor Keys. Application Cursor Keys send application program control functions. Normal Cursor Keys send ANSI cursor control sequences (up, down, left and right).
Exit	Exits the Setup panel and returns to on-line mode with the new settings.

On the right-hand side of the screen there is a list of languages. Select by cursor positioning the correct language for the keyboard being used.

At the bottom of the screen there is a ruler for setting Tabs. To set a tab, position the cursor on the ruler where the tab is to be set and press Enter. The position will change to show a T. To unset a tab, position the cursor on the T and press Enter.

Line mode operation (NVT)

Escape characters

The escape character can be used to access TELNET and issue TELNET commands. The default escape character is \$ (X'4A') and this can be overridden using the variable `lc_telnet_esc`. If the variable contains the value NOP, then the escape character processing is disabled.

If the escape character is entered at the end of a line without a following number, then the next string entered is concatenated to this string by omitting the ASCII carriage return and line feed control characters which are normally sent with each entered string; the escape character will not be sent.

For example:

```
1 2 3 4 5 6$    will send to the application    1 2 3 4 5 6 7 8
7 8
```

If the escape character entered is followed by one of the numbers listed below, then the escape character and the number are replaced with the appropriate ASCII character.

Escape no.	ASCII character
0	X'00' NUL
1	X'01' SOH
2	X'02' STX
3	X'03' ETX
4	X'04' EOT
5	X'05' ENQ
6	X'06' ACK
7	X'07' BEL
8	X'08' BS
9	X'09' HT
10	X'0A' LF
11	X'0B' VT
12	X'0C' FF
13	X'0D' CR
14	X'0E' SO
15	X'0F' SI
16	X'10' DLE
17	X'11' DC1
18	X'12' DC2

Escape no.	ASCII character
19	X'13' DC3
20	X'14' DC4
21	X'15' NAK
22	X'16' SYN
23	X'17' ETB
24	X'18' CAN
25	X'19' EM
26	X'1A' SUB
27	X'1B' ESC
28	X'1C' FS
29	X'1D' GS
30	X'1E' RS
31	X'1F' US
32 - 126	Standard ASCII characters X'20' - X'7E'
127	X'7F' DEL
128 - 255	Non standard 8 bit codes X'80' - X'FF' (See note below.)

Note Some of these codes have special meanings in TELNET protocol. Usage of some of these codes is described in 'Control codes and escape sequences' on page 362.

TELNET PF keys and client commands

The default function of the terminal control keys can be overridden by local variables as shown below.

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

Variable	Function	Key
LC_TELNET_KUKN	NOP	Unknown aid
LC_TELNET_KENT	SEND	ENTER send
LC_TELNET_KP01	@ISZCMD HELP	PF1 Session Manager help
LC_TELNET_KP02	TELNET	PF2 TELNET cmd
LC_TELNET_KP03	QUIT	PF3 Stop session
LC_TELNET_KP04	ISZCMD	PF4 Issue Session Manager command
LC_TELNET_KP05	NOP	PF5

Variable	Function	Key
LC_TELNET_KP06	@ISZCMD RETR	PF6 Retrieve cmd
LC_TELNET_KP07	BWD	PF7 Scroll backward
LC_TELNET_KP08	FWD	PF8 Scroll forward
LC_TELNET_KP09	HIDECMD	PF9 Hide command
LC_TELNET_KP10	TELNET IP	PF10 Issue TELNET IP
LC_TELNET_KP11	NOP	PF11
LC_TELNET_KP12	NOP	PF12
LC_TELNET_KP13	@ISZCMD HELP	PF13 Session Manager help
LC_TELNET_KP14	TELNET	PF14 TELNET cmd
LC_TELNET_KP15	QUIT	PF15 Stop session
LC_TELNET_KP16	ISZCMD	PF16 Issue Session Manager cmd
LC_TELNET_KP17	NOP	PF17
LC_TELNET_KP18	@ISZCMD RETR	PF18 Retrieve cmd
LC_TELNET_KP19	BWD	F19 Scroll backward
LC_TELNET_KP20	FWD	PF20 Scroll forward
LC_TELNET_KP21	HIDECMD	PF21 Hide command
LC_TELNET_KP22	TELNET IP	PF22 Issue TELNET IP
LC_TELNET_KP23	NOP	PF23
LC_TELNET_KP24	NOP	PF24
LC_TELNET_KOID	NOP	OID
LC_TELNET_KMAG	NOP	MAG
LC_TELNET_KPEN	NOP	PEN
LC_TELNET_KPA1	NOP	PA1
LC_TELNET_KPA2	NOP	PA2
LC_TELNET_KPA3	NOP	PA3
LC_TELNET_KATN	NOP	ATTN
LC_TELNET_KCLR	CLEAR	Clear data
LC_TELNET_KREQ	NOP	REQ
LC_TELNET_KSTR	NOP	STRUCT

Contents of each variable

The contents of each variable, which will be accessed when the key is pressed, should be one of the following actions:

SEND [*command*]

Sends *command* to the TELNET Server.

If no command is specified, the command in `t_command` is sent. The entered command will be echoed in an output line unless `s_telnet_hidec` is set to 'Y'. If 'More' is active, the display is scrolled forward one page. Also, if 'More' is active, null commands are not sent to the Server.

@SEND [*command*]

Same as SEND, except the command is not echoed in an output line.

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

HIDECMD

Hides the command until next SEND or @SEND.

The variable `s_telnet_hidec` will be set to 'Y' and the input, for example a password, subsequently entered in `t_command` will not be echoed in an output line.

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

QUIT

Terminates the session.

NOP

No operation.

CLEAR

Clears all queued, that is, unseen, output lines and resets scrolling to the latest line. 'More' mode is reset. If the BWD action is subsequently invoked, the last output lines will be displayed.

ISZCMD [*command*]

Issues Session Manager command *command* with the user's AUTH value.

If no command is specified, the command in `t_command` is used. The entered command will be echoed in an output line. Any message generated by the command processor can be displayed using `t_message` and will also be echoed in an output line.

@ISZCMD [*command*]

Same as ISZCMD except the command is not echoed in an output line.

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

ISZCMDA [*command*]

Same as ISZCMD except the command is issued with an AUTH of 9 rather than the user's AUTH value.

@ISZCMDA [*command*]

Same as ISZCMDA except the command is not echoed in an output line.

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

TELNET [*command*]

Issues the TELNET command *command*.

If no command is specified, the command in *t_command* is used. If no command is entered then the valid TELNET commands are displayed in an output line. The entered command is then echoed in an output line. If the command entered is not a valid TELNET command, an error message is displayed in an output line.

Valid TELNET server commands are:

AYT	Query connection.
AO	Abort output.
IP	Interrupt application.
SY[nch]	Clear data path.
Brk	Sends BREAK.

@TELNET [*command*]

Same as TELNET except the command is not echoed in an output line.

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

BWD [*number_lines_to_scroll*]

Scrolls back through the output lines.

If a number of lines to scroll is not specified, the value in *t_command* is used. If the value is 0 or non numeric, a page that is, the length of the panel's content, is scrolled.

The oldest history output line is identified as TOP OF OUTPUT and no scrolling beyond this line is possible. When there are more output lines to display, the lowest line contains MORE.

The maximum number of history lines by default is 60, but you can override this using the *ln_telnet_hist* variable. The value of this variable must be in the range 43-500.

FWD [*number_lines_to_scroll*]

Scrolls forward through the output lines.

If a number of lines to scroll is not specified, the value in `t_command` is used. If the value is 0 or non-numeric, a page that is, the length of the panel's content, is scrolled.

EBCDIC and ASCII translation (line mode and VTxxx mode)

TCP/TELNET communications are performed in ASCII, but Session Manager operates in EBCDIC. Session Manager uses two translate tables;

- one to translate from ASCII to EBCDIC, referred to as the OUTPUT table;
- the other to translate EBCDIC to ASCII, referred to as the INPUT table.

There are 256 entries in a translation table. For the INPUT table, each entry is the hexadecimal representation of an ASCII character. The corresponding EBCDIC character, as a hexadecimal value determines the position within the table of each ASCII character, starting from position 0 for the first entry.

For the OUTPUT table, each entry is the hexadecimal representation of an EBCDIC character. The corresponding ASCII character, as a hexadecimal value determines the position within the table of each EBCDIC character, starting from position 0 for the first entry.

By default Session Manager uses its own internal tables for this translation. If there are circumstances that require you to use different conversion rules, you can construct your own translation tables. To override the existing tables define a Session Manager TRANSTABLE with the name `#TCPxxxx`, where `xxxx` can be any character, and then identify the table in a local variable, named `lc_telnet_xtab`.

TN3270E operation

From Session Manager version 1.3.05 the TELNET client allows a URL specifying TN320E, in addition to the existing TELNET and TN3270 URL types.

When a TELNET session (`APPL = TCP_1`) specifies a TN3270E URL then during the TELNET client's negotiation with the remote TELNET server, the client will tell the server it supports TN3270E protocols. If the server also supports TN3270E protocols these will be used in place of the basic TN3270 protocols.

If TN3270E protocols are in use then if a non-blank device name is specified on the URL this will be passed to the TELNET server as the TN3270E device name using the `CONNECT` parameter.

The `CONNECT` value is typically used by a TN3270E server to either specify the virtual terminal name for the session or the name of a pool of virtual terminal names from which the server chooses the session's virtual terminal name.

The TN3270E client session operates and is configured in the same way as TELNET sessions.

- 1 To ensure Session Manager recognizes the session as a TN3270E client session, specify the `APPLID` parameter as `TCP_1`.
- 2 Specify the target TN3270E Server's URL (Uniform Resource Locator) in the `DATA` parameter in the following format:

```
TN3270E://[[userid][:password]@]host_IP_name|addr[:port_no]
[/device_name]
```

Note In this document, the 0x7C (that is, x'7C') character is always presented as the @ sign. It may be displayed as a different character in some non-English code pages. You should enter the appropriate 0x7C character symbol for the code page you are using.

The *userid* and *password* are 'advisory only' since all TN3270E clients, including Session Manager, cannot enter these fields automatically into the session. The values are contained in the session variables *s_telnet_user* and *s_telnet_pswd*. These could be entered on TN3270E Server signon screens using session scripts.

The *host_IP_name* or *addr* can be accessed using the variable *s_telnet_host*.

A port is an address in the range 0-32767. Each application has an individual port address. Standard applications, for example FTP, have pre-assigned port numbers and are known as 'well-known' ports. Ports for use by applications are assigned by the Portmap function of TCP/IP.

If the *port_no* is not specified, the value in variable *ln_telnet_port* is used. If this value is not specified, port 23 is used. The port number in use by the session can be accessed by the variable *s_telnet_port*.

The *device_name* can be made up of one or more variables by bracketing each variable name with ampersands. For example if the device name was in the ACB field, a URL might look like:

```
TN3270E://125:78:90:33/&s_acbname&
```

The maximum length for a device name is 8.

Read-only variables associated with TN3270E client support

These read-only variables are associated with TN3270E client support:

Variable	Description
<i>s_tn3270e</i>	If Session Manager has successfully negotiated with the TN3270 Server to use TN3270E protocols then this variable returns a value of 'Y'. Otherwise, and for all other terminal types, this variable returns a value of 'N'.
<i>s_tn3270e_dev</i>	Returns the device name, if any, supplied by the TN3270E Server at connection time.

Note The following printer-related TN3270E functions are not supported: ASSOCIATE, SCS-CTL-CODES and DATA-STREAM-CTL. The following optional TN3270E functions are not supported: RESPONSE, BIND-IMAGE and SYSREQ.

CHAPTER 15

Session Manager networking

The Session Manager networking system permits communication between tasks running on two or more Session Manager systems.

This chapter provides an explanation of some of the terms and concepts used when Session Manager is driving a multiple-system network. Users who only have a single Session Manager system need not read this chapter. The subjects covered in this chapter are:

- 'Overview' on page 374
- 'Establishing remote application sessions' on page 375
- 'User affinity' on page 383
- 'Specifying remote commands' on page 384
- 'Summary' on page 385

Overview

An overview of Session Manager networking can be found in ‘Using multiple Session Manager instances – networking’ on page 137. It is recommended that you review that information before proceeding with this chapter.

Sysplex users should also review the Parallel Sysplex support chapter in this manual.

Establishing remote application sessions

There are several reasons for establishing application sessions using a remote Session Manager:

- the session has a lower priority than the main session management processing. For example, some complex script sessions such as Application Builder could be run in another Session Manager. This second Session Manager would execute in a separate address space or partition which has a lower operating system dispatching priority than the main processing;
- the network load will be reduced for applications which run on remote machines. See the example below.

Reducing network load by using Session Manager networking

The following examples illustrate how network load can be reduced by using Session Manager networking.

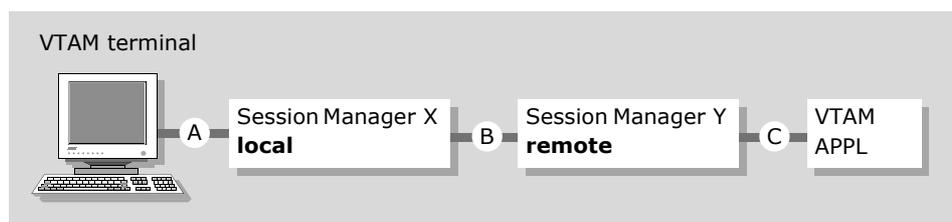
Conventional session path with a remote VTAM session



Session Manager network data reduction functions, datastream compression and MISER, can reduce the network traffic passing through path A but the network traffic passing through path B is not reduced.

Session path using a Session Manager remote session

Session Manager X is referred to as the *local* Session Manager, the Session Manager that the user is signed on to. Session Manager Y is referred to as the *remote* Session Manager, the Session Manager which connects to the application.



Session Manager network data reduction functions, datastream compression and MISER, can reduce the network traffic passing through both path A and B. Session Manager Y (the remote Session Manager) and the VTAM application are, ideally, executing in the same machine, in which case path C is totally within the machine and therefore there is no 'network' component in this path.

Provided the two Session Manager are connected using the Session Manager network, a Session Manager user would be unaware that one or more sessions are running remotely.

Data stream compression and data stream reduction can be applied to both paths A and B.

Remote sessions and data stream compression

To compress the outbound data streams on both paths A and B, either specify COMPRESS YES in the local definitions, or specify COMPRESS YES on the remote APPL definition. Compression is then processed in the remote Session Manager, Session Manager Y.

Remote sessions and MISER

The MISER feature is always enabled in Session Manager. The local session definitions, at Session Manager X, may specify MISER and can also specify RMISER. The MISER specification governs the inbound and outbound data streams from the local Session Manager, Session Manager X, on path A. The RMISER specification governs the inbound and outbound data streams from both Session Manager on paths B and A. Alternatively, in place of the RMISER definition, the remote APPL definitions in the remote Session Manager, Session Manager Y, may contain a MISER definition.

If MISER is not defined in the local Session Manager, even if RMISER is defined, the local Session Manager, Session Manager X, will issue Read Buffer commands when switching between sessions, processing commands, and so on.

Recommended MISER definitions for remote sessions

- 1 Specify in local Session Manager (no remote definition required)

```
MISER ERB                /* prevent Session Manager'
                          /*   generated RBUFs.

RMISER ON                /* full MISER processing, that is
outbound                 /*   inbound and outbound data flows
                          /*   and read command emulation.
```

Or

- 2 Specify in local Session Manager

```
MISER ERB                /* prevent Session Manager'
                          /*   generated RBUFs.
```

and on remote Session Manager APPL statement

```
RMISER ON                /* full MISER processing, that is
outbound                 /*   inbound and outbound data flows
                          /*   and read command emulation.
```

For more information on using MISER with remote sessions, refer to 'Network Data Minimiser (MISER)' on page 325.

The REMOTE session parameter

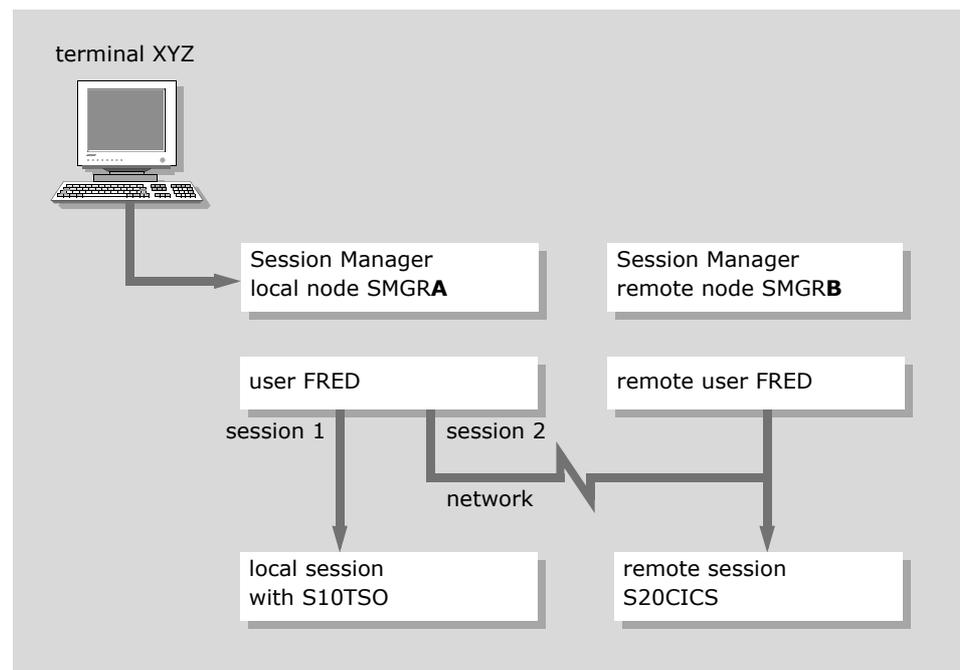
The REMOTE session parameter can be specified in the session definitions on the SYSTEM, PROFILE, USER, TERMINAL and APPL statements. It is specified:

```
REMOTE node-name
```

Examples

A user FRED signs onto his local Session Manager, node SMGRA, and his session 2 is to be a remote session initiated by another Session Manager, node SMGRB. Provided these two Session Manager are connected using a Session Manager network then the Session Manager administrator would define Fred's session 2 as follows:

```
USER fred
.....
SESSION 1 APPL S10TSO .....
.....
SESSION 2 APPL S20CICS REMOTE smgrb .....
.....
```



Session Manager QUERY command and remote sessions

Remote sessions, executing in the remote Session Manager, can be found using the Session Manager QUERY commands in the normal way. The remote user can only be found while an active remote session (or an active SEND command display) is executing. If the remote user has more than one remote session, they are grouped under one dummy user.

Using the previous example, the following messages would be returned if a 'Q User Fred' command were issued from the *remote* node, SMGRB:

```
ISZ0301I USER FRED SMGRA PROFILE PROF1 LU XYZ  
ISZ0320I (2) S20CICS SMGRB ACB S20TU004 LGMD D4B32795 STATUS-A- S( ),P( )
```

The following messages would be returned if a 'Q User Fred' command were issued from the *local* node, SMGRA:

```
ISZ0301I USER FRED SMGRA PROFILE PROF1 LU XYZ  
ISZ0320I (1) S10TSO SMGRA ACB S10TU002 LGMD D4B32795 -A- S( ),P( )  
ISZ0210I (2) S20CICS SMGRB ACB S20TU004 LGMD D4B32795 STATUS -A- S( ),  
P( ) Current
```

Script and exit execution sequence

The following diagram illustrates the order in which scripts and exit points are executed when a session is started with a remote application.

Local Session Manager	Remote Session Manager	Comments
(Node is set by the REMOTE configuration parameter)		
INITSCRIPT		
(decides which node by setting s_node)		
Session node is committed		Session 1
Exit point E31		
Session is initiated		User and session variables copied
Exit point E33 and Session 2	INITSCRIPT	
Note ENVIRONSCRIPT <i>not</i> run	Exit point E31 and Session 1	
STARTSCRIPT		Application is initiated
	Exit point E33 and Session 2	Variables copied back at completion of Environscript
	ENVIRONSCRIPT	
STARTSCRIPT	STARTSCRIPT	
• •		
AUTOSCRIPTS		
• •		
ENDSCRIPT		Application completes
	Exit point E39	
	TERMSCRIP	
Exit point E39		
TERMSCRIP		

Selection sequence for session parameter definition

Session parameters (except those outlined in ‘Exceptions in session parameter selection sequence’ below), are taken in two stages. Stage A is performed in the local Session Manager; Stage B is performed in the remote Session Manager.

Stage A

Local Session Manager session parameter priority, with highest shown first.

- 1 Parameters modified on the Menu screen or in the local Session Manager INITSCRIPT or in the E31 Exit or Exit script
- 2 SESSION parameters
- 3 USER session default parameters
- 4 PROFILE session parameters
- 5 PROFILE session defaults parameters
- 6 SYSTEM session defaults parameters
- 7 Local APPL parameters, if the session explicitly or implicitly identifies a local Session Manager APPL definition; that is, REFAPPL = YES (the default is REFAPPL=YES).

Stage B

Remote Session Manager session parameter priority, with highest shown first.

- 1 Stage A parameters
- 2 Remote APPL parameters, if the session explicitly or implicitly identifies a local Session Manager APPL definition; that is, REFAPPL=YES (the default is REFAPPL=YES).

Exceptions in session parameter selection sequence

Session ACB name

If the ACB name is not specified during points one to six in Stage A, it is *not* taken from a local Session Manager RANGE. The local parallel ACB is not used either. In either of these cases, if a remote Session Manager APPL is used, then the ACB can be taken from an associated remote Session Manager RANGE, or the remote parallel session ACB will be used.

Script processing

INITSCRIPT

If Stage A identifies an INITSCRIPT, this will run in the local Session Manager and therefore must be defined in the local Session Manager configuration file. This script will run before the session is identified as local or remote and so may alter the session’s node status using the `s_node` variable. If Stage B, the remote APPL definition, identifies an INITSCRIPT, this will run in the remote Session Manager and therefore must be defined in the remote Session Manager configuration file.

The INITSC command enables a suitably authorized user to disable or (re-)enable the running of the session 'initialization' script. For details, see the *Technical Reference*.

ENVIRONSCRIPT

If this script is identified in Stage A then it will *not* run on the local Session Manager at session start. However, if it is identified in the remote Session Manager, using the remote APPL definition, then it will run at session start and must be defined in the remote Session Manager configuration file. Any locally defined STARTSCRIPT will start after the remote ENVIRONSCRIPT has completed.

STARTSCRIPT

If this script is identified in Stage A, then it will run on the local Session Manager at session start and therefore must be defined in the local Session Manager configuration file. If it is identified in the remote Session Manager, using the remote APPL definition, then it will also run at session start and must be defined in the remote Session Manager configuration file. If a local script is defined then it will be started when the remote ENVIRONSCRIPT script has completed.

The STARTSC command enables a suitably authorized user to disable or (re-)enable the running of the session start script. For details, see the *Technical Reference*.

AUTOSCRIP and SAUTSEQ

These AUTO scripts will only run in the local Session Manager and therefore must be defined in the local Session Manager configuration file. These scripts can only be identified in Stage A; if any of these scripts are identified in Stage B, the remote APPL definition, they are ignored.

ENDSCRIPT

If Stage A identifies an ENDSCRIPT, it will run in the local Session Manager when the session is to end and therefore must be defined in the local Session Manager configuration file. If Stage B, the remote APPL definition, identifies an ENDSCRIPT, it is ignored.

TERMSCRIPT

If Stage B, the remote APPL definition, identifies a TERMSCRIPT this will run, when the session ends, in the remote Session Manager and therefore must be defined in the remote Session Manager configuration file. If Stage A identifies a TERMSCRIPT, this will run in the local Session Manager and therefore must be defined in the local Session Manager configuration file. This script will execute after the remote session has completed.

Variables and remote sessions

User defined variables at the user level (uc_ and un_) and session level (lc_ and ln_) are sent to the remote Session Manager when a remote session is started. When the session is initiated successfully, all session level variables are sent back from the remote Session Manager to the local Session Manager. Any user defined global level variables (gc_ and gn_) are not copied.

Script processing and remote sessions

If any of the scripts executing remotely experience either a 'script not found' or a 'DOMAX exceeded' condition, a system broadcast is *not* sent, (unlike a script running locally).

User affinity

When multiple Session Manager systems are being run, each one can be configured separately. This enables a user who spend the majority of their time using applications local to a particular Session Manager system, to be defined only on that Session Manager configuration file. If this user then uses a terminal connected to a remote Session Manager system, the remote Session Manager system passes the signon attempt to the user's local Session Manager system. This is known as User Affinity. It has two advantages:

- reduction in duplicate Session Manager definitions;
- a more direct path to the application, that is, instead of taking a diversion through the remote Session Manager (probably running on a different CPU) and then to the application, the new path goes directly to the local Session Manager, executing in the same VTAM machine as the application.

This is achieved using the variable `t_affinity` and the common enduser parameter `AFFINITY`.

Notes

- 1 The `t_affinity` variable can be set at signon and identifies to which Session Manager node the signon data is to be passed. If the Installation permits, that is, the variable is defined as an input field on the Signon panel, the user can identify to which Session Manager they wish to signon.
- 2 The `AFFINITY` parameter is used in the Session Manager configuration to define which Session Manager node is to be the Local Session Manager for each user.

A series of new parameters are passed to the Signon user exit point E21, to enable the exit point to decide or override the entered or configured Session Manager node for the user. No changes are required to the existing exits for `t_affinity` and `AFFINITY` to be actioned.

The *Technical Reference* manual contains details of the `AFFINITY` parameter; the *Panels, Scripts and Variables* manual contains details of the `t_affinity` variable; this manual contains details of the Session Manager user exit point E21.

Specifying remote commands

The SEND command can be used to send a command to a remote Session Manager, connected to the local Session Manager, using the Session Manager network.

The default authorization level required to invoke this command is 1. The command has the following format:

```
SEND nodename command
```

where:

nodename is a remote or local Session Manager node name. This must be the name specified on the LOCALNODE parameter in the Session Manager configuration file. The maximum length is 8 characters.

command is the command to be sent. If the command contains spaces, then it must be enclosed in delimiters, for example:

```
SEND node01 'query iszsmgr'
```

Note If any of the command parameters contain spaces, these must be enclosed in a different set of delimiters. Maximum length is 256 characters. Valid delimiters ')" !? (are explained in the *Technical Reference* manual.

The command is executed on behalf of the user in the remote Session Manager with an authority level of 0, unless an appropriate RUSER has been defined in the remote Session Manager. In that case, the authority level is taken from the RUSER definition. The RUSER must specify the originating node name and either the remote user name or a matching generic user name.

The following commands are valid for use with the SEND command:

```
BLOCK, BROADCAST, CLOSEDOWN, DELETE, DUMP, FLASH, FORCE, MSG,  
PASSFREE, SPIN, STARTLINK, STOP, STOPACB, STOPLINK, TERMINATE,  
TRACE, TTPSL, UPDATE, QTASK, QUERY.
```

The following commands can be invoked using SEND from a user's screen but not from the console:

```
DLOG, DSTORE, HELP.
```

The SEND command can invoke command scripts from the console or the menu. However, if ISZCMD is invoked with more than one full screen command, then only the first is processed; subsequent ones are ignored.

The following are the full screen commands which can be invoked with the SEND command:

```
DLOG, DSTORE, QTASK, HELP.
```

and all QUERY commands except:

```
QUERY SIGNON  
QUERY IDENT
```

All valid commands for these displays can be entered except those which cause another full screen display.

Summary

The following commands, statements, parameters and variables are required for the Session Manager Networking feature.

Command, statement, parameter or variable	Description
Networking feature	This feature must be enabled on the system before remote sessions are started; SEND commands are issued; sessions initiated from remote Session Manager are started.
LINK statement	Specifies links to other systems on the network.
LOCALNODE parameter	Specifies a nodename for the local Session Manager system. Each node on the network must have a unique node name defined. Entered on the SYSTEM statement.
REMOTE subparameter	Used on the SESSION parameter, to specify that the session is to start on a remote node. May be entered on the SYSTEM, PROFILE, USER, TERMINAL and APPL statements.
SEND command	Used to issue commands to a remote system on the network. Default authorization level 5.
t_tvnode variable	Identifies the nodename as specified on the LOCALNODE parameter.
t_node variable	Identifies the name specified by LOCALNODE in the system where the user is logged on (that is, some other system for remote sessions).
s_node variable	Identifies the name specified by LOCALNODE in the system where the session should start. The same as REMOTE specification on the session. This variable can only be updated before session start, for example, in menu processing, or Initscripts.

Parallel Sysplex support

This chapter describes the IBM Session Manager for z/OS facilities that support running of the product in a Parallel Sysplex environment.

The subjects covered in this chapter are:

- ‘Overview of facilities’ on page 388
- ‘Setting up for a Sysplex operation’ on page 392
- ‘VTAM application session recovery’ on page 398
- ‘VTAM application session recovery configuration’ on page 419
- ‘Sample signon validation exit script (E21)’ on page 422
- ‘Summary’ on page 424

Overview of facilities

Workload balancing

Session Manager is capable of supporting Parallel Sysplex Generic Resources (that is, a VTAM Generic ACB – see the *Technical Reference* for details of the `GENERICACB` parameter of the `SYSTEM` configuration statement). This capability enables Session Manager to balance the entire end user session load among its instances in a Parallel Sysplex environment based on the z/OS WLM (Workload Management) policies.

Sharing of the Session Manager configuration

In a Parallel Sysplex environment, and when operating with an OLA configuration, all Session Manager instances in the Parallel Sysplex environment can share the same configuration.

Sysplex configuration and operation

A number of Sysplex-related facilities are available to provide a single system image of multiple instances of Session Manager within a Session Manager Sysplex Group. These optional facilities are activated by the `SYSTEM` statement parameter `SYSPLXGroup` and its sub-parameters.

Note This section does not cover diagnostic sub-parameters that would only be specified at the request of your local representative. See the `SYSPLXGroup` description in the *Technical Reference* manual for a complete description of the sub-parameters.

The format of the `SYSPLXGroup` parameter, together with its main optional sub-parameters, is as follows:

```
SYSPLXGroup xxxx
  GLOBALMessages Yes|No|ON|OFF
  LOGSTREAMName logstreamname
  USERStructure Yes|No|ON|OFF
```

Typically an installation would specify just the `SYSPLXGroup` name and the Sysplex Group audit log, that is:

```
SYSPLXGroup xxxx
  LOGSTREAMName logstreamname
```

where

`xxxx` is the Session Manager Group name. This four-character name is used by all Session Manager instances that are to be members of this Group within the Sysplex.

`logstreamname` is the z/OS Sysplex Logger log stream name. This log stream name should be used by all members of the Group and is a common audit log that can be viewed by a user signed on to any member, using the Sysplex Summary and menu (see 'Sysplex Summary and menu' on page 389).

Sysplex-related facilities

By specifying `SYSPLXGroup` with just the `logstreamname` parameter the following facilities are activated:

Sysplex networking

Sysplex networking is an alternative to Session Manager VTAM LU0 LINK statements and does not require any LINK statements to be defined. Sysplex networking employs z/OS Cross-system Coupling Facility (XCF). See ‘Sysplex Networking’ on page 392 for more details.

Automatic user reconnection within a Sysplex

This facility automatically reconnects users to their original active back-end sessions after they have been disconnected from a Session Manager instance. See ‘User reconnection’ on page 395 for details.

It replaces the script-based mechanism described in ‘Sample signon validation exit script (E21)’ on page 422 and allows users connected to Session Manager via VTAM terminals, or pseudo-VTAM terminals established by external Telnet servers, to reconnect to their original active sessions. The facility employs the z/OS Coupling Facility (CF) Sysplex services for data sharing (XES) to maintain a central User List structure in the Coupling Facility containing currently signed on users and disconnected users for all Session Manager instances within the Group. The structure name is the chosen `SYSPLXGroup` name prefixed with `ISMUSER_`.

Caution: This facility relies on the data in the User List CF structure and therefore if this list becomes unavailable and you have not enabled System Managed Duplexing, then a user who reconnects/signs on may be logged on again as a new user, creating a second, duplicate, user within the Session Manager Sysplex environment.

Sysplex Group wide BROADCAST and MSG commands

Messages sent with the `BROADCAST` and `MSG` commands are sent to all members of the Group. For example, if a `BROADCAST` is sent with the `ALL` parameter then all users, on all members of the Group, will receive the broadcast. Similarly, for example, if a `MSG` command is sent to a particular user then even if the user is signed on to another member in the Sysplex Group the user will receive the message and the issuer is not required to know which member in the Group the target user is signed on to. This ‘global’ scope uses Session Manager’s Sysplex networking. See the descriptions of the `BROADCAST`, `MSG`, `QUERY BROADCAST`, `DELETE MSG` and `DELETE BROADCAST` commands in the *Technical Reference* manual.

Sysplex Group audit log

If `logstreamname` is specified on the `SYSPLXGroup` parameter then this member will write all its audit messages to the log stream. Other members of the Group would also be configured to write to the same log stream, allowing the viewer to display a combined audit log from all the members of the Group (see ‘Sysplex Summary and menu’ below).

Sysplex Summary and menu

The Sysplex menu provides a user who is signed on to any member in the Sysplex Group with a single-system view of all the members within the Sysplex Group. This view allows the user to display details and issue commands at the Group level as well as at a particular Group member level. If `logstreamname` is specified on the

SYSPLXGroup parameter the menu includes a facility to view the Sysplex Group audit log. The viewer allows scrolling of the log, searches, and tailoring of the display to suit the user's requirements. This facility is described in detail in the *User and Administrator* manual.

RUSER statements are not required

If SYSPLXGroup is specified then although RUSER statements can be specified to control the authority that a remote user has on the local instance, the RUSER definitions are not required. If no appropriate RUSER statement is found then the remote user's authority will be the same as was established when the user signed on. For nodes in a SYSPLXGroup any matching RUSER statements will override the user's AUTH and OLAClass, therefore any redundant RUSER statements should be deleted.

Note Even if it is envisaged that most users would not use the SEND command to issue remote commands, bear in mind some of the Sysplex menu functions utilize the SEND command to issue remote commands on behalf of the user. If SYSPLXGroup is not specified and no appropriate RUSER statement is found then the remote user's authority will be minimal.

OnLine Administration (OLA)

To avoid I/O contention, the OLA facility operates in only one instance. If SYSPLXGroup is not specified, an installation has to nominate a specific instance where OLA processing is performed, by defining an APPLID on the OLA APPL definition. However, when SYSPLXGroup is specified the APPLID should be removed because Sysplex networking automatically nominates an instance that is active, and that instance will be accessed via a remote OLA session. If the APPLID parameter is not removed, the nominated instance will continue to be used.

In a Sysplex environment where the multiple Session Manager instances are assigned to the same SYSPLXGroup it is mandatory that all the Session Manager instances must share the same configuration datasets. Also any given SYSPLXGroup must *not* have a mix of Classic and OLA Session Manager instances.

The majority of parameters in each Session Manager Group member's SYSTEM statement are identical. So to improve maintainability a common SYSTEM statement containing the identical parameters has been introduced (ISZSYSCM) leaving the few member-specific parameters in each member's SYSTEM statement.

Virtual Terminal Masking (VTM) maintenance

To avoid I/O contention, the VTM maintenance facility operates in only one instance. If SYSPLXGroup is not specified, an installation has to nominate a specific instance where VTM maintenance processing is performed by defining an APPLID on the VTM PROFILE. When SYSPLXGroup is specified the APPLID should be removed because Sysplex networking automatically nominates an instance that is active, and that instance will be accessed via a remote VTM maintenance session. If the APPLID parameter is not removed, the nominated instance will continue to be used.

In a Sysplex environment where the multiple Session Manager instances are assigned to the same SYSPLXGroup it is mandatory that all the Session Manager instances must share the same configuration datasets.

TPSL function SPLXLOCUSER

If SYSPLEXGroup is specified, this function can be used by a script to locate which node, if any, a user is signed on to. Having established the node, the script can then issue various commands concerning the user via the Session Manager SEND command. See the SPLXLOCUSER description in the *Panels, Scripts and Variables* manual for more details.

TPSL function SPLXLOG

If *logstreamname* is specified on the SYSPLEXGroup parameter, this function can be used by a script to access the common audit log. See the SPLXLOG description in the *Panels, Scripts and Variables* manual for more details.

TPSL function SPLXNODES

If SYSPLEXGroup is specified, this function can be used by a script to identify which nodes are active and for each node details such as the node's job name, the z/OS instance, the node's ACB name and the node's status. See the SPLXNODES description in the *Panels, Scripts and Variables* manual for more details.

VTAM application session recovery

Session Manager provides VTAM application session recovery in two modes:

- Planned Session Manager to Session Manager switch
- Session recovery after catastrophic failure.

These features only fully apply to users who access Session Manager using VTAM terminals, or to TN3270 client users who use an independent TN3270 Server providing access to Session Manager using VTAM virtual terminals. Users connected to Session Manager via Session Manager's own 3270 Server will be disconnected from Session Manager during a planned switch.

See 'VTAM application session recovery' on page 398.

Selecting the facilities required

If none of the above facilities is required then do not specify the SYSPLEXGroup parameter on the SYSTEM statement. In some cases it is possible to turn off an individual facility that is not required, although Sysplex Networking is mandatory if the SYSPLEXGroup parameter is specified.

The following functions are optional:

Automatic user reconnection within a Sysplex. This can be turned off by specifying USERStructure NO on the SYSPLEXGroup parameter of the SYSTEM statement.

Sysplex Group audit log. This can be turned off simply by not specifying *logstreamname* on the SYSPLEXGroup parameter of the SYSTEM statement.

Sysplex Group wide BROADCAST and MSG commands. The scope of Broadcasts and Messages can be limited to the current instance by specifying GLOBALMessages NO.

Setting up for a Sysplex operation

Sysplex networking

There are no specific external setup requirements for Session Manager Sysplex networking. The volume of traffic will vary depending on an installation's use of remote commands, remote sessions, BROADCAST commands and MSG commands. Bear in mind that some of the Sysplex Menu functions will issue remote commands and OLA/VTM maintenance sessions can be remote. There will always be a burst of activity when each member starts or stops.

Automatic user reconnection within a Sysplex

The CF user structure must be defined in the Coupling Facility's CRFM policy using the program IXCMIAPU. For more details see 'Networking and Sysplex considerations (if required)' on page 127.

Sysplex Group audit log

The CF log stream structure will have to be defined in the Coupling Facility's CRFM policy using the program IXCMIAPU and the log stream must be defined to the z/OS Logger also, using the program IXCMIAPU. For more details see 'Networking and Sysplex considerations (if required)' on page 127.

Sysplex menu

The NODE PDSE has to be defined to each Session Manager instance's started task JCL. For Classic users, sample JCL is in ISZCSTRT. For OLA users, sample JCL is in ISZCOLA. For each member's SYSTEM statement add the INITIAL_CMD parameter with a value of ISZSPNOD.

Note If the INITIAL_CMD parameter already specifies a value then either edit the named script and add a CALL ISZSPNOD or edit ISZSPNOD and add a CALL xxxxxx where xxxxxx is the named script and make the INITIAL_CMD value ISZSPNOD.

Sysplex Networking

When each member of the Sysplex Group initializes it will automatically connect directly to all the other active members in the Group. Once a connection is established between two members, a virtual Session Manager LINK is created using Session Manager Sysplex networking.

Sysplex networking connections are functionally identical to connections established by Session Manager VTAM LU0 LINK statements and so explicit LINK statements are not required to connect the members in the Group. And indeed, if explicit LINK statements are defined to connect members they will automatically be deactivated and replaced by the virtual Sysplex network LINK. See 'LINK statements in a Sysplex' on page 393 for more details on LINKs in a Sysplex.

The Sysplex networking virtual LINKs can be queried by the Query Net LINK command just like conventional VTAM LU0 LINKs. However, the virtual LINKs cannot be stopped or restarted.

Sysplex networking is used by the following functions:

- Monitoring node status

- Remote commands, that is the SEND command
- Remote sessions
- BROADCAST and MSG commands
- SPLXNODES TPSL function
- Sysplex menu via the SEND command and the SPLXNODES TPSL function
- OLA and VTM maintenance via remote sessions.

LINK statements in a Sysplex

When Session Manager is operating in a Sysplex; that is, when SYSPLEXGroup has been specified, and no LINKs need to be established with instances outside the Group then no explicit Session Manager VTAM LU0 LINKs need be defined. This is because Session Manager Sysplex networking will automatically establish and maintain virtual LINKs with all other members of the Group. If either an explicit LINK statement that connects to another member of the Group is activated, or an explicit LINK statement with the same name as another member's LOCALNODE is activated, then these explicit LINK statements will be automatically deactivated and replaced by the virtual Sysplex LINK.

If an instance that executes outside of the Sysplex needs to be connected to a member in the Sysplex Group, you need to define a VTAM LU0 LINK statement as usual. However, there are some restrictions with this type of connection. When a non SYSPLEXGroup Session Manager node connects to another non SYSPLEXGroup node, via Session Manager VTAM LU0 LINKs, each node can access other non SYSPLEXGroup nodes connected to the other's node without connecting directly to them using LINK statements. However, when a connection is made between a SYSPLEXGroup node and a non SYSPLEXGroup node then other members' nodes connected to the SYSPLEXGroup node are not accessible via this link.

Also, to connect pre-1.3.15 Session Manager nodes to 1.3.15 or higher nodes, a PTF must be applied to the pre-1.3.15 nodes. Contact your local representative for details.

Connecting nodes

The definitions required to connect nodes vary depending on how the nodes are linked. There are basically four ways to configure the connection of multiple nodes, and other combinations can be extrapolated from these.

The four basic types are:

Nodes within a Sysplex Group

This is where all the nodes to be connected are within the same Sysplex. This type of connection only requires the SYSPLEXGROUP parameter on the SYSTEM statement of each node to be specified with the same group name. No explicit LINK statements are required because virtual LINK statements will be automatically created.

Nodes outside a Sysplex Group

This is where all the nodes to be connected do not have the SYSPLEXGROUP parameter specified but can be connected via an SNA connection. This type of connection requires sufficient explicit VTAM LU0 LINK statements to be defined for the program to establish a path between all the nodes. A path would consist of either a direct connection using in a pair of LINK statements (one at each end) or an indirect connection using LINK pairs to one or more intermediate nodes.

Nodes within and outside a Sysplex Group

The Sysplex nodes (the nodes specifying the same SYSPLEXGROUP name) require no further definitions to connect to each other. For each non-Sysplex node to connect to all the Sysplex nodes a LINK pair, one in the Sysplex node and one in the non-Sysplex node, is required for each Sysplex node. This is because Sysplex nodes do not operate as intermediate nodes.

For example:

If there were four nodes in a Sysplex Group named PLX1, PLX2, PLX3 and PLX4, and they were to be connected to a node outside the Sysplex named Z1, then the following LINK definitions would be required:

PLX1	PLX2	PLX3	PLX4	Z1
LINK LZ1				LINK LPLX1
	LINK LZ1			LINK LPLX2
		LINK LZ1		LINK LPLX3
			LINK LZ1	LINK LPLX4

Note Be aware that although remote commands and remote sessions will operate across all these nodes, other Sysplex facilities will not operate outside the Sysplex Group.

Nodes in two Sysplex Groups

The Sysplex nodes for each group require no further definitions to connect to the other nodes in their group. However, to connect all the nodes from both groups together, as above, then a pair of LINK statements is required for each connection.

For example:

If there were two nodes in the GA Sysplex Group named GA1 and GA2, and two nodes in the GB Sysplex Group named GB1 and GB2, then the following LINK definitions would be required:

GA1	GA2	GB1	GB2
LINK LGB1		LINK LGA1	
LINK LGB2			LINK LGA1
	LINK LGB1	LINK LGA2	
	LINK LGB2		LINK LGA2

Note Be aware that although remote commands and remote sessions will operate across all these nodes, other Sysplex facilities will not operate outside each Sysplex Group.

User reconnection

In a Parallel Sysplex environment with at least one Session Manager instance per image, terminal sessions are workload balanced across the Session Manager instances using VTAM Generic Resources.

When a user disconnects, or after a network failure between the user's VTAM terminals and a particular Session Manager instance, Session Manager maintains the user's active back-end sessions. When the user(s) sign on again to Session Manager they may be routed to a different Session Manager instance which has no record of these 'hanging' back-end sessions, so the user(s) receive 'ALREADY SIGNED ON' messages when they (re-)access their applications.

In such situations, if a user is already signed on to (or disconnected from) a particular Session Manager node and the user attempts to sign on again, two mechanisms are provided to transfer the sign-on to the original Session Manager node, enabling the user to continue to access the existing active back-end sessions.

The two mechanisms are the 'Automatic user reconnection within a Sysplex' on page 389 and the original (pre-1.3.15 Session Manager) 'Sample signon validation exit script (E21)' on page 422. Although two mechanisms are provided, automatic user reconnection within a Sysplex should be used if all the nodes are executing within the same Sysplex Group. This mechanism provides improved performance during user sign-on and, once set up, the Group requires no changes if new members join or leave. It is selected via the `USERStructure` sub-parameter of the `SYSPLXGroup` and is on by default.

The second mechanism is implemented using a sample E21 exit script – for details, see 'Sample signon validation exit script (E21)' on page 422. To migrate from the script-based mechanism to the `USERStructure` mechanism see 'User reconnection migration' below.

User reconnection migration

This section discusses how an installation can migrate from the original 'sample signon validation exit script (E21)' mechanism, `ISZE21GA`, to 'automatic user reconnection within a Sysplex'.

The easiest way to migrate is in a single set of steps, as follows:

- 1** Define the structure. The structure name is the chosen `SYSPLXGroup` name prefixed by `ISMUSER_`. For more details see 'Networking and Sysplex considerations (if required)' on page 127.
- 2** Ensure that the members of the Sysplex Group are at Session Manager version 1.3.15 or higher, if not, upgrade them by closing all the systems, performing the upgrade and then (for OLA installations only) restarting one member to enable you to perform the next step.
- 3** For OLA installations, use OLA to update the common `SYSTEM` definitions in member `ISZSYSCM` as follows. If the `SYSPLXGroup` parameter is not specified then add it. If the `USERStructure` sub-parameter is specified then remove it.

- 4 For Classic installations, make the following change to all members' SYSTEM statements. If the SYSPLEXGroup parameter is not specified then add it. If the USERStructure sub-parameter is specified then remove it.
- 5 Closedown all the members in the Sysplex.
- 6 If you are running the original 'sample signon validation exit script (E21)' mechanism, ISZE21GA, remove the E21 exit script from the OPTION statement in the ISZCONxx member by deleting the E21 S parameter value. If you are running any E21 script other than ISZE21GA or a variant then E21 S should remain. For OLA installations the OPTION statement is in the CONFIG PDSE.
- 7 Remove all LINK statements from all members' configuration (other than those that connect to Session Manager systems outside the Sysplex). For OLA installations start one member and simply delete all LINK statements.
- 8 Restart all members of the Sysplex Group.

However, if an installation wishes to migrate using a phased approach then follow the appropriate path below. These paths have been designed to minimize the number of steps involved. The path to follow depends on whether your systems are Classic or OLA systems.

Classic systems

- 1 Define the structure. The structure name is the chosen SYSPLEXGroup name prefixed by ISMUSER_. For more details see 'Networking and Sysplex considerations (if required)' on page 127.
- 2 For each member in turn
 - a Ensure that the member is at Session Manager version 1.3.15 or higher, if not upgrade it by closing the member and then perform the upgrade.
 - b If the SYSPLEXGroup parameter is not specified on the member's SYSTEM statement then add it. If the USERStructure sub-parameter is specified then remove it.
 - c If the member is executing then close it.
 - d Restart the member.
- 3 With all members now executing both the E21 script based mechanism and the 'automatic user reconnection within a Sysplex' mechanism, we can now remove the older mechanism with all the members active.
- 4 For each member in turn:
 - a If you are running the original 'sample signon validation exit script (E21)' mechanism, ISZE21GA, remove the E21 exit script from the OPTION statement in the ISZCONxx member by deleting the E21 S parameter value. If you are running any E21 script other than ISZE21GA or a variant then E21 S should remain. For OLA installations the OPTION statement is in the CONFIG PDSE.
 - b Remove all LINK statements (other than those that connect to Session Manager systems outside the Sysplex).
 - c Issue a "UPD E21 E" command to dynamically stop the E21 script exit from processing. You could use the SEND command to issue the command on all the other remote members; that is, SEND xxxx "UPD E21 E" where xxxx is the LOCALNODE name of the remote system.

OLA systems

- 1** Define the structure. The structure name is the chosen SYSPLEXGroup name prefixed by ISMUSER_. For more details see ‘Networking and Sysplex considerations (if required)’ on page 127.
- 2** For each member in turn starting with the nominated OLA member:
 - a** Ensure that the member is at Session Manager version 1.3.15 or higher, if not, upgrade it by closing the member and then perform the upgrade.
 - b** For the nominated OLA member only. Start the member if it is not executing. Sign on to it and using OLA update the common SYSTEM definition in member ISZSYSCM as follows. If the SYSPLEXGroup parameter is not specified then add it. If the USERStructure sub-parameter is specified then remove it.
 - c** Close the member if it is executing.
 - d** Restart the member.
- 3** With all members now executing both the E21 script based mechanism and the ‘automatic user reconnection within a Sysplex’ mechanism we can now remove the older mechanism with all the members active.
- 4** For each member in turn
 - a** If you are running the original ‘sample signon validation exit script (E21)’ mechanism, ISZE21GA, remove the E21 exit script from the OPTION statement in the ISZCONxx member by deleting the E21 S parameter value. If you are running any E21 script other than ISZE21GA or a variant then E21 S should remain. For OLA installations the OPTION statement is in the CONFIG PDSE.
 - b** For the first member only, delete all LINK statements (other than those that connect to Session Manager systems outside the Sysplex).
 - c** Issue a “UPD E21 E” command to dynamically stop the E21 script exit from processing. You could use the SEND command to issue the command on all the other remote members; that is, SEND xxxx “UPD E21 E” where xxxx is the LOCALNODE name of the remote system.

VTAM application session recovery

Overview

Session Manager provides VTAM application session recovery (also known as High Level Availability or HLA) in two modes, as summarized in ‘VTAM application session recovery’ on page 391.

Planned switches This facility allows an installation to perform planned maintenance on a Session Manager instance, on z/OS, or on its associated z/Series with minimum disturbance to Session Manager users. An operator command (SWITCHplex) enables all users to be transferred from one Session Manager instance to another.

An application session is defined with a recovery level of HIGH, INTERMEDIATE or NONE (for details see ‘Recovery levels’ on page 412). Those sessions defined as HIGH or INTERMEDIATE will be maintained across the transfer.

The user’s terminal sessions will usually be maintained across the transfer. If a user is active in a recovered session then they should be unaware of the switch. If the user is active in a session that is not recovered then they will be returned to the main menu. However, if the subsequent planned maintenance affects components in the user terminal paths, then the affected users may become disconnected, requiring them to reconnect to the Session Manager group.

Similarly, if the planned maintenance affects components between Session Manager and the application then these application sessions may be broken, in which case the users will have to re-establish the application sessions.

Session recovery This provides the ability to transfer automatically all the users from one Session Manager instance to another Session Manager instance when either Session Manager, z/OS or its associated z/Series suffers a catastrophic failure.

Only application sessions defined with a recovery level of HIGH or INTERMEDIATE will be recovered, but the extent of the recovery will depend on which components in the Session Manager group are impacted. Potentially, in the best case, all HIGH and INTERMEDIATE application sessions may be recovered. All terminal sessions to the failing Session Manager instance will be broken, requiring the users to reconnect to the Session Manager group and gain access to their recovered application sessions.

Recommended scenario

To implement a Session Manager environment to cater for both planned switches and session recovery involves a performance overhead. If session recovery after a catastrophic failure is not required we recommend the following procedure for planned switches, that is, maintenance:

A Session Manager Controller instance is required and this Controller and the active Primary instances must be in the same Sysplex and must be using the same Session Manager SYSPLEXGROUP.

Maintenance on Primary instances

- 1 When maintenance is required on one or more active Primary instances, or on the z/OS system or on the hardware, it is recommended that on a machine not affected by the maintenance you start one or more Standby instances that are targeting the active Primary instances that will be affected by the maintenance.
- 2 Issue the SWITCHPLX command (we recommend that you use the Sysplex Summary and Menu facility) to transfer all the users and their HIGH and INTERMEDIATE sessions. The active Primary instances will terminate and the Standby instances will now be acting as the active Primary instances.
- 3 Perform the required maintenance.
- 4 Restart the original active Primary instances as Standby instances for the original Standby instances.
- 5 Issue the SWITCHPLX command to transfer all the users and their HIGH and INTERMEDIATE sessions back to the original instances.

The original Standby instances will terminate and the original active Primary instances will again act as active Primary instances.

Maintenance on Controller instances

When maintenance is required on the Controller instance, or on the z/OS system or on the hardware, it is recommended that you start a Standby Controller instance on a machine not affected by the maintenance.

Note Only RECOVERYLEVEL HIGH sessions can be recovered in this procedure; RECOVERYLEVEL INTERMEDIATE sessions will be lost.

- 1 Issue the SWITCHPLX command (we recommend that you use the Sysplex Summary and Menu facility) to transfer all the HIGH session ACBs to the Standby Controller instance. The Controller instance will terminate and the Standby Controller instance will now be acting as the Controller instance.
- 2 Perform the required maintenance.
- 3 Restart the original Controller instance as the Standby Controller instance for the original Standby Controller instance.
- 4 Issue the SWITCHPLX command to transfer all the HIGH session ACBs back to the original Controller instance.

The original Standby Controller instance will terminate and the original Controller instance will now act as the Controller.

For further details please read the FAQs below, 'Principles and components' on page 410 and 'VTAM application session recovery configuration' on page 419.

FAQs

Q: Do I need to set-up VTAM application session recovery?

A: VTAM application recovery is only required if access to any VTAM applications is critical to the your operation during a planned Session Manager to Session Manager switch for maintenance or during a catastrophic failure. See 'Overview' on page 398.

Q: What applications are recovered?

A: Only connections to VTAM applications are recoverable. Session Manager will not recover the VTAM applications but just the connections. Therefore it is the customer's responsibility to have the necessary recovery in place for their VTAM applications after a catastrophic failure. TCP/IP sessions and Session Manager internal sessions (For example: Online Administration (OLA), System Management Menu (SMM), and any Command scripts) are not recovered.

Q: Are Session Manager internal sessions recoverable?

A: Session Manager internal sessions (For example: Online Administration (OLA), System Management Menu (SMM), and any Command scripts) are not recovered.

Q: Are Remote sessions recoverable?

A: Remote sessions to or from the active Primary instance are not recovered.

Q: How are ACB ranges affected by planned switches and session recovery?

A: When a session is started it will use an ACB from the specified range. When this session is transferred to a Standby instance the ACB is also transferred. Therefore if the original instance is restarted the ACB will not be available from the range as it is allocated to the Standby instance. The ACB will become available again when all sessions using the ACB on the Standby instance are terminated.

Q: Can I share ACB ranges between nodes?

Session Manager nodes within a High Level Availability Sysplex environment must not use the same ACBs. It is recommended to share the configuration across nodes in a Sysplex environment but if ACBs are specified on the shared RANGE statements then they must be unique.

One possible set-up to overcome this situation would be to add to the SYSTEM statement:

```
%% let gc_acb_suffix = 'cc'
```

(where cc is unique for each SYSTEM statement) and to change your RANGE statements to include this variable, for example:

```
RANGE CICS
FROM ISZ%%gc_acb_suffix%%001
TO fff
HEX
```

Q: Under what conditions are VTAM application sessions recovered?

A: Session Manager provides VTAM application recovery in two modes:

- 1 A planned Session Manager to Session Manager switch when, for example, a Session Manager instance needs to be closed down for maintenance.
- 2 After a catastrophic failure to the Session Manager environment.

See 'Planned switches' and 'Session recovery' on page 398.

Q: Do I have to recover all VTAM applications?

A: No, you can set three different recovery levels:

- 1 HIGH – This option will perform the following where possible:
 - a Recover forward sessions to this application if the owning Session Manager node fails or the Session Manager Controller fails.
 - b Transfer the forward sessions to the node’s Session Manager Standby during a planned Session Manager to Session Manager switch.
 - c Transfer the forward sessions to the Session Manager Standby Controller during a planned Session Manager Controller to Session Manager Standby Controller switch.
- 2 INTERMEDIATE – This option will perform the following where possible:
 - a Recover forward sessions to this application if the owning Session Manager node fails.
 - b Transfer the forward sessions to the node’s Session Manager Standby during a planned Session Manager to Session Manager switch.

Note INTERMEDIATE sessions will not be recovered if the Session Manager Controller fails or during a planned Session Manager Controller to Session Manager Standby Controller switch.
- 3 NONE – This is the default. With this option forward sessions to this application will not be recovered if the Session Manager node fails, and these sessions are not transferred during a planned Session Manager to Session Manager switch.

Note However these sessions will be unaffected and therefore still active if a Session Manager Controller fails or during a planned Session Manager Controller to Session Manager Standby Controller switch, as the session does not interact with the Session Manager Controller or Session Manager Standby Controller.

See ‘Recovery levels’ on page 412.

Q: How do I define High, Intermediate and None applications?

A: The RECOVERYLEVEL parameter, specified on the Session Manager APPL statement, determines the level of recovery for sessions to the application defined by the APPL statement. This can also be specified on the USER, PROFILE or SYSTEM statement. See the *Technical Reference* for details.

See ‘Recovery levels’ on page 412.

Q: What do I need to define in order to recover High recovery level applications?

A: A Controller, a Standby Controller and a Standby instance are all required. These instances need to be in the same Sysplex and the same SYSPLEXGROUP as the active Primary instance. Defining the APPL with recovery level HIGH will cause the Session Manager Controller to use the VTAM MNPS facility on these forward sessions, to enable the Session Manager Standby Controller to recover the sessions if the Session Manager Controller fails. The Session Manager Controller (rather than the Session Manager instance) will perform the VTAM open of the virtual terminal ACBs relating to the application session; consequently, the VTAM ACB definitions must be available to the Controller’s VTAM.

This option will force the Session Manager MISER function to be used on each application session.

Specify HIGH on the RECOVERYLEVEL parameter on the APPL statement for each application that you wish to recover. The recovery level can also be set as a common end user parameter on the USER, PROFILE or SYSTEM statement. See the *Technical Reference* for details.

See 'Principles and components' on page 410.

Q: What do I need to define in order to recover Intermediate recovery level applications?

A: A Controller and a Standby instance are all required. A Standby Controller is not required. These instances need to be in the same Sysplex and the same SYSPLEXGROUP as the active Primary instance.

Note If the Session Manager Controller fails, the forward sessions will not be recovered. The Session Manager Controller Standby is not used.

This option will cause the Session Manager Controller to be used (rather than the Session Manager instance) for the VTAM open of the virtual terminal ACBs relating to the application session. Consequently, the VTAM ACB definitions must be available to the Controller's VTAM.

This option will force the Session Manager MISER function to be used on each application session.

Specify INTERMEDIATE on the RECOVERYLEVEL parameter on the APPL statement for each application that you wish to recover. The recovery level can also be set on the USER, PROFILE or SYSTEM statement. See the *Technical Reference* for details.

See 'Recovery level INTERMEDIATE' on page 412.

Q: What do I need to define in order to recover None recovery level applications?

A: This is the default. With this option forward sessions to this application will not be recovered if the Session Manager node fails, and these sessions are not transferred during a planned Session Manager to Session Manager switch. If all APPL statements have a recovery level of NONE then the Session Manager Standbys and the Session Manager Controllers are not required, as no sessions are recoverable.

NONE is the default recovery level setting, however NONE can be specified on the RECOVERYLEVEL parameter on the APPL statement for each application that you do not wish to recover. The recovery level can also be set on the USER, PROFILE or SYSTEM statement. See the *Technical Reference* for details.

Q: When is the recovery level established for a session?

A: The recovery level for a session is established when the ACB is opened. If, for example, a previous session has used the same ACB as new session (even if it was a different application) and that previous session caused the ACB to open, then the new session's RECOVERYLEVEL will be the value established when the ACB was opened.

Q: Can a session's recovery level change?

A: The SYSTEM statement CLOSEACBINACT YES parameter causes ACBs to close when there are no active sessions using the ACB and if the ACB is subsequently used by a session then the ACB will be re-opened. The default setting for CLOSEACBINACT is NO which causes the ACB to remain open until the instance closes. Consequently CLOSEACBINACT YES may be used to re-assign the recovery level applicable to sessions using a particular ACB. See the *Technical Reference* for details.

Q: How and where do I define a Controller instance?

A: The Controller is defined by specifying SYSPLEXTYPE C on the Controller's System statement. See the *Technical Reference* for details.

Ideally, if resources allow, the Session Manager Controller should reside on separate hardware and on a different z/OS system from the Standby Controller, the active instances and their Standbys. However the Controller should be on the same Sysplex as the Standby Controller, the active instances and their Standbys. They can reside on the same hardware and z/OS but if the hardware or z/OS is the cause of the problem then recovery may not be possible.

Q: How and where do I define a Standby Controller?

A: The Standby Controller is defined by specifying SYSPLEXTYPE S on the Standby Controller's System statement. See the *Technical Reference* for details.

Ideally, if resources allow, the Session Manager Standby Controller should reside on separate hardware and on a different z/OS system from the Controller, the active instances and their Standbys. However the Standby Controller should be on the same Sysplex as the Controller, the active instances and their Standbys. They can reside on the same hardware and z/OS but if the hardware or z/OS is the cause of the problem then recovery may not be possible.

Q: How and where do I define a Standby instance?

A: The Standby instance is defined by specifying STANDBY *nodename* on the Standby instance's SYSTEM statement, where *nodename* is the target node name of the active instance for this Standby instance. STANDBY N is the default setting, which would indicate that this instance is not a Standby for any active instances. See the *Technical Reference* for details.

Ideally, if resources allow, the Session Manager Standby instance should reside on separate hardware and on a different z/OS system from the Controller, the Standby Controller and the active instance. However the Standby instance should be on the same Sysplex as the Controller, the Standby Controller and the active instance. They can reside on the same hardware and z/OS but if the hardware or z/OS is the cause of the problem then recovery may not be possible.

Q: How and where do I define an active instance?

A: The active instance is defined by specifying SYSPLEXTYPE I on the active instance's SYSTEM statement. SYSPLEXTYPE N is the default setting, so if you wish to use RECOVERYLEVEL HIGH and RECOVERYLEVEL INTERMEDIATE sessions you must specify SYSPLEXTYPE I. See the *Technical Reference* for details.

Ideally, if resources allow, the Session Manager active instance should reside on separate hardware and on a different z/OS system from the Controller, the Standby Controller and the Standby instance. However the active instance should be on the same Sysplex as the Controller, the Standby Controller and the Standby instance. They can reside on the same hardware and z/OS but if the hardware or z/OS is the cause of the problem then recovery may not be possible.

Q: What is the preferred order when starting up the Controller, Standby Controller, the active instances and Standby instances?

A: Ideally the Controller, then the Standby Controller, then the Standby instances and then the active instances. When the Standby Controller is started it will synchronize with the Controller instance. When the Standby instance is started it will synchronize with the active instance.

However they may be started in any order - see below.

Q: What happens if they are started out of turn?

A: The Standby Controller needs to know the current state and all the activity of the Controller. Therefore if the Standby Controller is started before the Controller then when the Controller is started it will synchronize with the Standby Controller.

The Standby instance needs to know the current state and all the activity of the active instance. Therefore if the Standby instance is started before the active instance then when the active instance is started it will synchronize with the Standby instance.

Q: What happens if the Controller or Standby Controller is not active?

A: If no Controller is active then you cannot initiate any HIGH or INTERMEDIATE recovery level sessions (their ACBs cannot be opened).

If no Standby Controller is active then there is no recovery available for any HIGH recovery level sessions if the Controller becomes unavailable.

However if the Controller is not active when the Standby Controller initiates it will wait for a defined period of time, as defined on the STANDBYTAKETIME sub-parameter on the SYSPLEXGROUP parameter on the Standby Controller's System Statement, after which it will assume the role of the Controller. If the Controller is started after the Standby Controller has taken on the role of the Controller then the Controller will take on the role of the Standby Controller.

Q: What happens when the Controller fails due to either an environment failure (hardware or software) or the Controller was terminated after the SWITCHPLX command was issued on the Standby Controller?

A: If a Standby Controller is active it will assume the role of the Controller and where possible any HIGH recovery level sessions will be recovered. Any INTERMEDIATE recovery level sessions will not be recovered. Any NONE recovery level sessions will not be affected as they do not interact with the Controller.

After a SWITCHPLX command has transferred the sessions from the Controller to the Standby Controller, the Controller will automatically close down, with Return Code = 272.

In the event of a failure, if a Standby Controller is not active then no sessions will no recovered. Also no sessions will be recoverable until a Controller or a Standby Controller is initiated.

Q: What happens if the Controller is restarted after the Standby Controller assumes the role of the Controller?

A: The Controller will assume the role of the Standby Controller and synchronize with the active Controller.

Q: What happens when the Controller is terminated using the CLOSEDOWN command?

A: Any open ACBs for sessions defined as HIGH or INTERMEDIATE recovery level will be terminated and sessions defined as HIGH or INTERMEDIATE can not be initiated. Initiate a Controller if you require access to these sessions.

Q: What happens when the Standby Controller is terminated using the CLOSEDOWN command?

A: As there is now no Standby Controller active if the Controller fails then no sessions are recoverable. The SWITCHPLX command is also not available to transfer recovery level HIGH sessions. Initiate a Standby Controller if you require recovery level HIGH sessions to be recovered if the Controller fails or if you need to use the SWITCHPLX command.

Q: What happens if the Standby instance is not active?

A: If the Standby instance is not active and the active Primary instance fails, no sessions are recoverable. Configure and initiate a Standby instance if you require sessions to be recoverable. The SWITCHPLX command is also not available to transfer recovery level HIGH and INTERMEDIATE sessions.

Q: What happens when the active instance fails due to either an environment failure (hardware or software) or the active instance was terminated after the SWITCHPLX command was issued on the Standby instance?

A: If a Standby instance is active it will assume the role of the active instance and where possible any HIGH or INTERMEDIATE recovery level sessions will be recovered. Any NONE recovery level sessions will not be recovered.

After a SWITCHPLX command has transferred the sessions from the active instance to the Standby instance, the active instance will automatically close down, with Return Code = 276.

If a Standby instance is not active then no sessions will be recovered.

Q: What happens when the active instance was terminated using the CLOSEDOWN command?

A: As all the users and their sessions will be terminated during the closedown process there are no sessions to be recovered or transferred to the Standby instance.

Q: What happens if the active instance is restarted after the Standby instance assumes the role of the active instance?

A: The active instance will be available for any new users to log on to either explicitly or indirectly via generic ACBs and the Workload manager. If a Standby instance was defined it will continue to act as the active instance for the users and sessions that it recovered and will assume the role of a Standby instance for any users and their sessions on the restarted active instance. Users and their sessions that were recovered by the Standby instance are not transferred back to the restarted active instance.

Q: What is the preferred order when shutting down the Controller, Standby Controller, the active instances and the Standby instances?

A: The preferred order is the active instances, then the Standby instances, then the Standby Controller, and then the Controller.

However they may be shutdown in any order - see below.

Q: What happens if they are shut down out of turn?

A: If the Controller is shutdown first then any open ACBs for sessions defined as HIGH or INTERMEDIATE recovery level will be terminated.

If the Standby Controller is shutdown before the Controller then in the event of a Controller failure no HIGH recovery level sessions are recoverable.

If the Standby instance is shutdown before the active instance then no sessions are recoverable.

Q: What do you mean by hot standby?

A: When initiated the Controller and Standby Controller will synchronize and the active instances and their Standby instances will synchronize. The Standby Controller will mirror any activity on the Controller and the Standby instances will mirror any activity on their active instances. Therefore after a failure or controlled SWITCHPLX the Standby Controller will immediately assume the role of the Controller and the Standby instances will assume the role of the active instances.

This mirroring process will add an additional workload to all the instances.

Q: How do I transfer VTAM application sessions from an active instance to its Standby so that I can perform maintenance?

A: The SWITCHPLX command can be used to transfer all the HIGH and INTERMEDIATE recovery level sessions to the Standby instance. After the sessions have been transferred the active instance will then closedown, with Return Code = 276. It is recommended that the Sysplex Summary and Menu facility is used to enter the SWITCHPLX command.

Q: How do I transfer VTAM application sessions from the Controller to the Standby Controller so that I can perform maintenance?

A: The SWITCHPLX command can be used to transfer all the HIGH recovery level sessions to the Standby Controller. After the sessions have been transferred the Controller will then closedown, with Return Code = 272. It is recommended that the Sysplex Summary and Menu facility is used to enter the SWITCHPLX command.

Q: Do the users need to logon again to an active instance after sessions have been transferred to a Standby instance after the SWITCHPLX command has been issued?

A: Users who access an active instance using VTAM terminals or TN3270 client users who use an independent TN3270 Server that provides access to the active instance using VTAM virtual terminals will not have to log on again as their terminal session to the active instance will, if possible, be transferred to the Standby instance. Users connected to the active instance via Session Manager's own TN3270 Server will be disconnected from the active instance during the SWITCHPLX process and not transferred to the Standby instance. Therefore these users will need to reconnect to an active instance. However any sessions that were recovered by the Standby instance will still be active.

Q: Do the users need to logon again to an active instance after sessions have been transferred to a Standby instance due to an unplanned failure?

A: Yes, all terminal sessions will be disconnected and therefore these users will need to reconnect to an active instance. However any sessions that were recovered by the Standby instance will still be active.

Q: Can I use the Controller as an active Session Manager instance?

A: No, the Controller cannot be used as an active instance and no recovery level HIGH or INTERMEDIATE sessions can be started.

Q: Can I use the Standby Controller as an active Session Manager instance?

A: No, the Standby Controller cannot be used as an active instance and no recovery level HIGH or INTERMEDIATE sessions can be started.

Q: Can I use the Standby instance as an active Session Manager instance?

A: Yes the Standby instance can be used as an active Session Manager instance. However you need to consider the following consequences:

- a** When either the Standby instance or the instance it is standing by for are initiated then the Standby instance must synchronize with the instance. This will put an additional workload on the Standby instance.
- b** The Standby instance will mirror all activity on the active instance so that it is in a position to immediately take over all of the instance's users and their sessions. This will put an additional workload on the Standby instance.
- c** When the active instance for which this instance is a Standby for either fails or is switched then this Standby instance will then be in control of all the users and their sessions from the now inactive instance. This will put an additional workload to on the Standby instance.

Therefore, if resources allow, the Standby instance should ideally not be used as an active Session Manager instance.

Note You may wish to isolate the Standby Instance so that it will only be used actively when it needs to recover its target's users and sessions. You can do this by configuring it **not** to use the GENERICACB that is in use by the other instances. This would prevent the Workload Manager from scheduling users to sign on to it both during its standby mode and after recovery.

Q: If an active instance can also be a Standby instance how do you recommend I define my Standby instances?

A: Ideally, for optimum resilience to failures, the Session Manager Standby should operate on different hardware to its partner. So, for example, you could arrange each active instance to be backed up by its Standby instance, forming a ring. That is, instance A has instance B as its Standby; instance B has instance C as its Standby; instance C has instance D as its Standby; and instance D has instance A as its Standby.

Q: When is MNPS utilized?

A: When a failure occurs in a active Session Manager instance, or when a planned Session Manager to Session Manager SWITCHPLX is performed, all recovery level HIGH and recovery level INTERMEDIATE application sessions are recovered without recourse to VTAM's MNPS session capabilities.

However, a different mechanism operates in the unlikely event that the Session Manager Controller within the Sysplex fails or, more likely, that it is required to transfer the Session Manager Controller's workload to its own Standby Controller after the SWITCHPLX command is issued. Then all Session Manager instances within the Sysplex group are affected, and only those application sessions defined with recovery level HIGH will be recovered. These sessions will be recovered using VTAM's MNPS support. Application sessions defined with a recovery level of INTERMEDIATE will not be recovered. Any NONE recovery level sessions will not be affected as they do not interact with the Controller.

By operating in this way Session Manager allows an installation to manage the use of MNPS and therefore the corresponding additional resources required to maintain the MNPS sessions, including the deployment of the Session Manager Standby Controller.

As an example, an installation might regard its production CICS application sessions as critical, and worth the extra MNPS resources, so it sets the recovery level as HIGH. The installation may regard its TSO and test CICS sessions as less critical, and therefore not worth the extra MNPS resources, but would still like them to be recovered when performing a planned Session Manager to Session Manager SWITCHPLX command, or to be recoverable after a Session Manager instance failure, so sets the recovery level to INTERMEDIATE.

To utilize MNPS the Controller and the Standby Controller must use different VTAMs. However, if they use the same VTAM then SNPS will be used. With SNPS if the Controller fails the ACBs and sessions will be transferred to the Standby Controller. If VTAM or the z/OS system fails the ACBs and sessions will not be recovered.

Q: Are there any Session Manager MNPS options I should be aware of?

A: The PSTIMER parameter specifies the time, in seconds, that data required for session recovery will be retained by VTAM after a Controller failure. This should be set to a value that allows the Standby Controller enough time to recover the sessions.

Q: Are there any VTAM MNPS settings I should be aware of?

A: For sessions to have MNPS fully enabled then PERSIST=MULTI must be specified on the VTAM definition for session ACBs, as in:

```
S09KT VBUILD TYPE=APPL
S07KT* APPL ACBNAME=S07KT*,          *
      AUTH=(ACQ,PASS,NVPACE),        *
      PARSESS=YES,                   *
      EAS=10,                         *
      PERSIST=MULTI,                  *
      SRBEXIT=YES,                    *
      VPACING=0
```

Specifying PERSIST=MULTI by itself does not cause MNPS to be used. MNPS will only be used when both PERSIST=MULTI and recovery level HIGH are specified.

Q: What messages are issued by the Controller?

A: The Controller will issue messages to the log indicating that it is the Controller, when it starts and completes synchronizing with the Standby Controller and when it starts and completes transferring sessions to the Standby Controller.

A complete audit log of all the messages issued by all the Session Manager instances can be viewed via the Sysplex Summary and Menu facility.

Q: What messages are issued by the Standby Controller?

A: The Standby Controller will issue messages to the log indicating that it is the Standby Controller, when it starts and completes synchronizing with the Controller and when it starts and completes transferring sessions from the Controller.

A complete audit log of all the messages issued by all the Session Manager instances can be viewed via the Sysplex Summary and Menu facility.

Q: What messages are issued by the Standby instance?

A: The Standby instance will issue messages to the log indicating that it is the Standby instance, when it starts and completes synchronizing with the active instance and when it starts and completes transferring sessions from the active instance, when each user signs on and off the target active Primary instance, and when a user on the target active Primary instance starts and stops a recovery level HIGH or INTERMEDIATE session.

A complete audit log of all the messages issued by all the Session Manager instances can be viewed via the Sysplex Summary and Menu facility.

Q: What messages are issued by the active instance?

A: The active instance will issue messages to the log indicating that it is the active instance, when it starts and completes synchronizing with the Standby instance and when it starts and completes transferring sessions to the Standby instance.

A complete audit log of all the messages issued by all the Session Manager instances can be viewed via the Sysplex Summary and Menu facility.

Q: How do I find out about the current status of the Controller, the Standby Controller, the Standby instances and the instances?

A: The 'Display nodes' option in the Sysplex Summary and Menu facility will display the current status of all the nodes in the Session Manager's Sysplex Group, including any defined Controller, Standby Controller, active instances and Standby instances.

Q: How do I find out about the current number of active users and their VTAM sessions on the Controller, the Standby Controller, the Standby instances and the instances?

A: The 'Display nodes' option in the Sysplex Summary and Menu facility will display a list of all the nodes in the Session Manager's Sysplex Group, including any defined Controller, Standby Controller, active instances and Standby instances. Issue a QUERY command against the required instance to display additional information.

Q: Are there any limitations or restrictions?

A: Any SPY, DEMO or VIEW will be terminated if the user is signed on to the failing instance or after a Session Manager to Standby Session Manager switch. The WINDOWS facility will also be terminated under these circumstances. Remote sessions defined as Recovery Level HIGH or INTERMEDIATE can not be started. The synchronization for any sessions that are running scripts will be delayed until the session script completes. The session will not be recovered until the script and the synchronization for this session completes. If the session has a long-running script then the session may never be synchronized and therefore will not be recovered after a switch or a failure.

Recovery Level HIGH or INTERMEDIATE sessions can not be started with the NLOG command.

Please contact your local support representative for further details.

Q: Are there any performance considerations?

A: The recovery process uses hot standby technology which allows for immediate recovery of sessions defined as recovery level HIGH and INTERMEDIATE, and also utilizes MNPS for sessions defined as recovery level HIGH and INTERMEDIATE. This involves synchronization and the constant mirroring of activity between any active instances and their Standbys and the Controller and its Standby. All this additional processing will greatly increase your CPU usage.

Please contact your local support representative for further details.

Principles and components

The session recovery facility uses z/OS VTAM Multi-Node Persistent Sessions (MNPS) in conjunction with Session Manager Sysplex components that create a resilient wrapper around the existing Session Manager nodes.

It operates on a 'hot standby' basis. This provides a more timely recovery than starting a backup Session Manager after a failure and then establishing the internal task structure representing the failed instance's users and their sessions.

The following address spaces, which must be in the same Sysplex and the same Session Manager SYSPLEXGROUP, are used:

Session Manager instance This is an existing Session Manager node and typically in a Sysplex there would be a number of these. These instances must be configured as instances capable of using RECOVERYLEVEL HIGH and RECOVERYLEVEL INTERMEDIATE application sessions by specifying SYSPLEXTYPE I. Note that the default for SYSPLEXTYPE is N.

Session Manager Standby Each Session Manager instance can also be configured to be a Session Manager Standby which is a hot standby for another Session Manager instance. The standby immediately takes over from its partner when its partner fails. The failure could be a failure in the Session Manager instance's hardware, a failure in the Session Manager instance's z/OS or an unrecoverable failure in Session Manager itself. These instances must be configured as instances capable of using RECOVERYLEVEL HIGH and RECOVERYLEVEL INTERMEDIATE application sessions by specifying SYSPLEXTYPE I. Note that the default for SYSPLEXTYPE is N.

The Session Manager Standby will also take over from its partner when a planned Session Manager to Session Manager switch is performed. Once the standby has taken over from its partner then if the standby's instance has its own standby (a third node) then these recovered users will now belong to the standby's instance and become backed up by the third node.

Ideally, for optimum resilience to failures, the Session Manager Standby should operate on different hardware to its partner. So, for example, you could arrange each instance to be backed up by its standby, forming a ring. That is, instance A has instance B as its standby; instance B has instance C as its standby; instance C has instance D as its standby; and instance D has instance A as its standby.

Session Manager Controller For applications defined as recoverable, the Session Manager Controller controls all the Session Manager instances' VTAM interactions. And for applications defined with the highest recoverability, the application sessions will be established as MNPS sessions. The Controller cannot double up as a Session Manager instance.

Session Manager Standby Controller This is a hot standby for the Session Manager Controller. This immediately takes over from the Controller when the Controller fails, or in a planned takeover when the Controller must be closed down.

Only the sessions of applications defined with the highest recoverability (the MNPS sessions) will be maintained across the takeover. Sessions of applications that are not defined with the highest recoverability will not be maintained across this takeover and if no applications are defined with the highest recoverability then the Session Manager Standby Controller is not required. Ideally, for optimum resilience to failures, the Session Manager Standby Controller should operate on different hardware to its partner.

The 'Sysplex Summary and menu facilities' chapter of the *User and Administrator* manual gives details of displaying nodes and standby status.

Recovery levels

The RECOVERYLEVEL parameter, specified on the Session Manager APPL statement determines the level of recovery for sessions to the application defined by the APPL statement. This can also be specified on the USER, PROFILE or SYSTEM statement. See the *Technical Reference* for details. The parameter has the following values:

Recovery Level HIGH

This option will perform the following where possible:

- a Recover forward sessions to this application if the owning Session Manager node fails or the Session Manager Controller fails.
- b Transfer the forward sessions to the node's Session Manager Standby during a planned Session Manager to Session Manager switch.

This option will use the Session Manager Controller to provide this support. The option will cause the Session Manager Controller to use the VTAM MNPS facility on these forward sessions, to enable the Session Manager Standby Controller to recover the sessions if the Session Manager Controller fails.

This option will cause the Session Manager Controller to be used (rather than the Session Manager instance) for the VTAM open of the virtual terminal ACBs relating to the application session. Consequently, the VTAM ACB definitions must be available to the Controller's VTAM.

This option will force the Session Manager MISER function to be used on each application session.

Recovery level INTERMEDIATE

This option will perform the following where possible:

- a Recover forward sessions to this application if the owning Session Manager node fails.
- b Transfer the forward sessions to the node's Session Manager Standby during a planned Session Manager to Session Manager switch.

This option will use the Session Manager Controller to provide this support.

Note If the Session Manager Controller fails, the forward sessions will not be recovered. The Session Manager Controller Standby is not used.

This option will cause the Session Manager Controller to be used (rather than the Session Manager instance) for the VTAM open of the virtual terminal ACBs relating to the application session. Consequently, the VTAM ACB definitions must be available to the Controller's VTAM.

This option will force the Session Manager MISER function to be used on each application session.

Recovery level NONE

This is the default. With this option forward sessions to this application will not be recovered if the Session Manager node fails, and these sessions are not transferred during a planned Session Manager to Session Manager switch. If all APPL statements have a recovery level of NONE then the Session Manager Standbys and the Session Manager Controllers are not required, as no sessions are recoverable.

The 'Sysplex Summary and menu facilities' chapter of the *User and Administrator* manual gives details of displaying standby status.

SESACB settings Sessions started with the SESACB session must either associate the session with an APPL statement which has the RECOVERYLEVEL specified or use RECOVERYLEVEL on the USER statement.

Application granularity

Some CICS and IMS systems provide a variety of functions. Some of these functions could be considered critical and some less so. As the recovery level setting can only operate at the application level (see note below), an installation must make a judgement call on which recovery level applies to the application.

One approach could be to create more than one APPL for the application, and define a separate session on the user's PROFILE for each APPL. You would define a different STARTSCRIPT on each APPL that will navigate to the appropriate functions. The appropriate recovery level value could then be defined on each APPL statement.

Note In fact, although the recovery level is specified on the Session Manager APPL statement, it is implemented at the virtual terminal or ACB level and then only when the ACB is opened. Normally, Session Manager will only open the ACB once, and only close it at Session Manager close down.

However, the CLOSEACBINACT parameter causes Session Manager to close the ACB when there are no sessions active, and if the ACB is subsequently re-used it will be re-opened. Consequently, be aware when sharing an ACB or ACB ranges across applications, that if the first application that causes an ACB to open has a different recovery level to an application that subsequently uses it, then the first application's recovery level will be used.

ACB and RANGE statements

Session Manager nodes within a High Level Availability Sysplex environment must not use the same ACBs. It is recommended to share the configuration across nodes in a Sysplex environment but if ACBs are specified on the shared RANGE statements then they must be unique.

One possible set-up to overcome this situation would be to add to the SYSTEM statement:

```
%% let gc_acb_suffix = 'cc'
```

(where cc is unique for each SYSTEM statement) and to change your RANGE statements to include this variable, for example:

```
RANGE CICSH
FROM ISZ%%gc_acb_suffix%%001
TO fff
HEX
```

Recovery planning

Managing use of VTAM MNPS

With the Session Manager Sysplex wrapper implemented, when a failure occurs in a Session Manager node, or when a planned Session Manager to Session Manager switch is performed, all recovery level HIGH and recovery level INTERMEDIATE application forward sessions are recovered without recourse to VTAM's MNPS session capabilities.

However, a different mechanism operates in the unlikely event that the Session Manager Controller within the Sysplex fails or, more likely, that it is required to transfer the Session Manager Controller's workload to its own Standby Controller. Then all Session Manager nodes within the Sysplex group are affected, and only those forward sessions of applications defined with recovery level HIGH will be maintained across the takeover. These forward sessions will be recovered using VTAM's MNPS support. Forward sessions of applications defined with a recovery level of either INTERMEDIATE or NONE, will not be maintained across the takeover.

By operating in this way Session Manager allows an installation to manage the use of MNPS and therefore the corresponding additional resources required to maintain the MNPS sessions, including the deployment of the Session Manager Standby Controller.

As an example, an installation might regard its production CICS application sessions as critical, and worth the extra MNPS resources, so it sets the recovery level as HIGH.

The installation may regard its TSO and test CICS sessions as less critical, and therefore not worth the extra MNPS resources, but would still like them to be transferred when performing a planned Session Manager to Session Manager switch, or to be recoverable after a Session Manager node failure, so sets the recovery level to INTERMEDIATE.

Placement of the Controllers

This section discusses the deployment of the Session Manager Controller and the Session Manager Standby Controller. In most cases this design allows an installation to maintain the structure of its existing implementations of Session Managers in a Sysplex.

For application forward sessions established with recovery level HIGH, and for recovery level INTERMEDIATE applications, the Session Manager instances do not perform VTAM interactions directly. Instead, these interactions are performed on the Session Manager instances' behalf by the Controller.

If an installation intends to specify recovery level HIGH for one or more of its applications, and therefore imply the use of MNPS sessions, then the installation needs to be aware of the following requirements when recovering MNPS sessions.

VTAM MNPS sessions

Since only the Session Manager Controller (and its Standby) interacts with MNPS, only the Session Manager Controller (and its Standby) has to conform to MNPS requirements. The main requirements concerning Session Manager are

- 1 VTAM MNPS support is only available in a Sysplex.
- 2 For an ACB to be eligible to support MNPS it must be defined to VTAM with the PERSIST=MULT parameter on the VTAM APPL statement.

- 3 The Session Manager Controller and the Session Manager Standby Controller must reside in two different z/OS instances as they must not use the same VTAM. However, the two z/OS instances must be within the same Sysplex; and if subplexing is used then the Controllers must be within the same subplex.
- 4 In order to support VTAM MNPS the Session Manager Controller, its Standby and the target applications must use the z/OS VTAM High Performance Routine option set to Rapid Transport Protocol (HPR=RTP) paths. See the *SNA Network Implementation Guide* for details of the MNPS network requirements.
- 5 Only sessions that connect to applications on other z/OS HPR=RTP VTAMs can be recovered. That is, local applications on the Session Manager Controller (or its Standby) VTAM, cannot be recovered. Consequently the placement of the Session Manager Controller and the Session Manager Standby Controller within its mainframe complex must be in a z/OS instance that does not contain any recovery level HIGH applications.

The Session Manager Controller's z/OS and its Standby's z/OS must reside within the same Sysplex as the Session Manager instances.

Session Manager Standby Controller

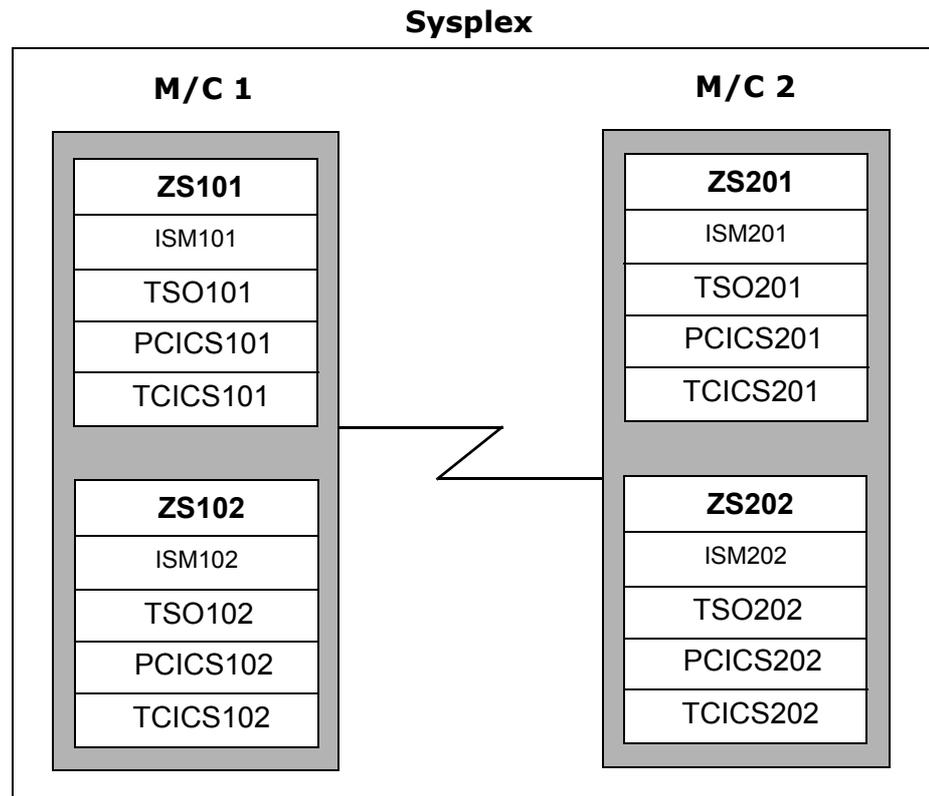
If at least one recovery level HIGH application is defined then the Session Manager Standby Controller must be deployed. However, if no recovery level HIGH applications are defined then the Session Manager Standby Controller is not required.

When the Session Manager Standby Controller takes over from the Session Manager Controller it will use the MNPS FORCETKO (Force Takeover) option rather than the Planned Takeover technique. Therefore the Session Manager Controller will use the FORCETKO=ALL parameter when opening its VTAM ACBs.

Example recovery configuration

Sample original layout

The following diagram shows a Sysplex without session recovery:

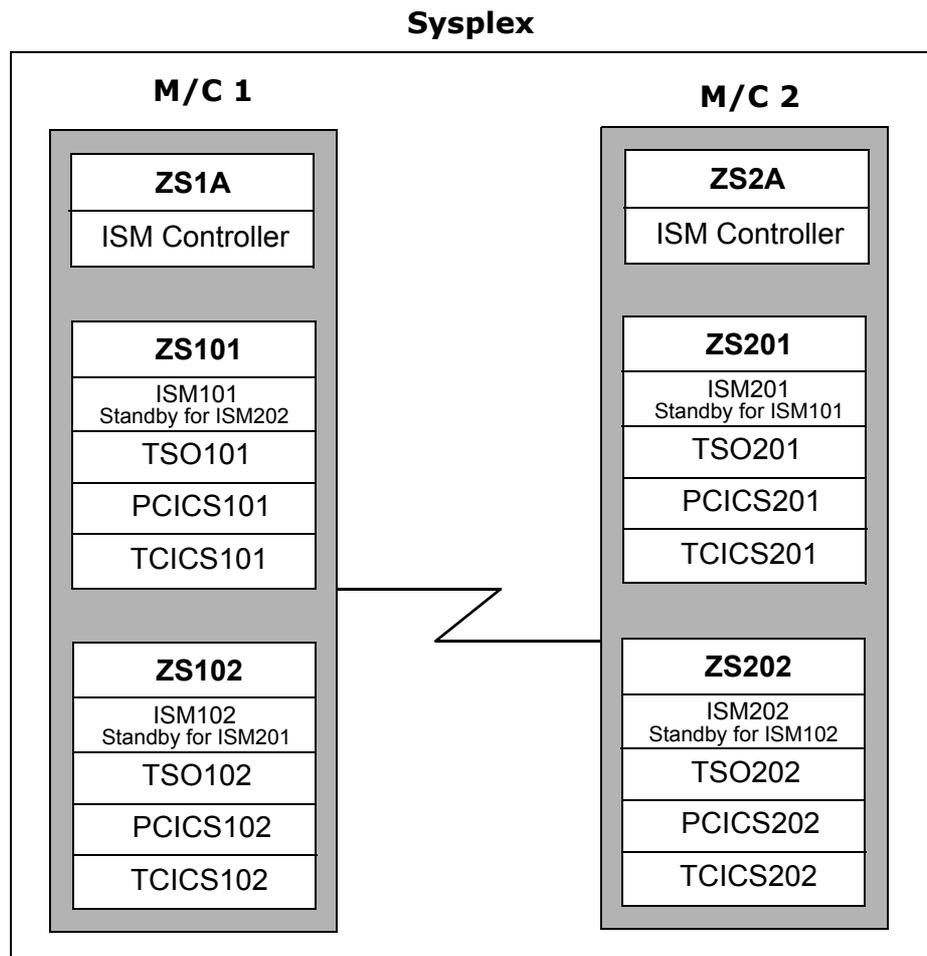


Normally this is not the whole picture as installations typically configure their session managers to enable users to access all, or a substantial subset of their many applications. Therefore, potentially, the vast majority of the applications accessed could be outside this Session Manager Sysplex.

Typically the z/OS Workload Manager will select a member of an application generic ACB group that is local to the initiator of the session, in preference to any other member. So, for example, if a user on ISM202 initiated a session with the production CICS applications (PCICS101, PCICS102, PCICS201 and PCICS202) via their common generic ACB of PCICS then the Workload Manager would typically select the local instance PCICS202.

Sample recovery layout

The following diagram shows a Sysplex with session recovery deployed:



This layout assumes there is at least one recovery level HIGH application and therefore shows the Session Manager Standby Controller deployed.

Remember, a number of the applications accessed by the Session Manager users could be outside the Sysplex shown; and these applications may include recovery level HIGH and recovery level INTERMEDIATE applications.

When an Session Manager user initiates a session request to a recovery level HIGH or recovery level INTERMEDIATE application, the VTAM request is issued by the Session Manager Controller. If the request is for a generic ACB then the Workload Manager will select a member of the generic ACB group, based on the performance parameters of each member, as there is no local member. In this case, all other things being equal, the Workload Manager's choice will typically be the member with the least number of active sessions. Therefore this technique will introduce true workload balancing, although at a cost of a longer network path compared to the original layout.

Note that since the Session Manager Controller now issues the request to start a session, if local application names are used then they will be local to the Controller rather than the Session Manager instance. For example, if you use the local name for the TSO application, typically this would be TSO, rather than the application's network name, say S35TSO. Then the TSO local to the Session Manager instance will not be used: instead you will start a session with the TSO local to the Controller.

This is unlikely to be a big issue, as network names would normally be used. However, if these local names are used, the session (a local session) will not be able to be recovered by MNPS, since the application uses the same VTAM as the Controller. This effectively downgrades a recovery level HIGH to a recovery level INTERMEDIATE. To prevent this, use the VTAM APPL's network name, that is, the APPL definition's label name.

VTAM application session recovery configuration

More information on all the Session Manager configuration parameters specified below can be found in the *Technical Reference* manual.

How to configure a Controller

On the SYSTEM statement for the Controller:

- Set SYSPLEXTYPE to C.
- Set STANDBY to N (this is the default).
- Set SYSPLEXGROUP to your Session Manager Sysplex group suffix (up to 4 characters). The name will be prefixed with ISM to form a Sysplex name of ISMxxxx.

How to configure a Standby Controller

On the SYSTEM statement for the Standby Controller:

- Set SYSPLEXTYPE to S.
- Set STANDBY to N (this is the default).
- Set SYSPLEXGROUP to your Session Manager Sysplex group suffix (up to 4 characters). The name will be prefixed with ISM to form a Sysplex name of ISMxxxx.

How to configure a Standby Instance

On the SYSTEM statement for the Standby Instance:

- Set SYSPLEXTYPE to I (the default is N).
- Set STANDBY to the node name of the target instance for this Standby.
- Set SYSPLEXGROUP to your Session Manager Sysplex group suffix (up to 4 characters). The name will be prefixed with ISM to form a Sysplex name of ISMxxxx.

How to configure a Primary Instance

On the SYSTEM statement for the Primary Instance:

- Set SYSPLEXTYPE to I (the default is N).
- Set STANDBY to N (this is the default).
- Set SYSPLEXGROUP to your Session Manager Sysplex group suffix (up to 4 characters). The name will be prefixed with ISM to form a Sysplex name of ISMxxxx.

How to configure a Primary Instance which is also a Standby Instance

On the SYSTEM statement for the Primary Instance:

- Set SYSPLEXTYPE to I (the default is N).
- Set STANDBY to the node name of the target instance for this Standby.
- Set SYSPLEXGROUP to your Session Manager Sysplex group suffix (up to 4 characters). The name will be prefixed with ISM to form a Sysplex name of ISMxxxx.

How to set the session recovery level

It is recommended that the recovery level is set on the APPL statement:

Set RECOVERYLEVEL to HIGH or INTERMEDIATE or NONE (the default).

However the recovery level may also be specified as a Common end user parameter on the SYSTEM, PROFILE, USER or TERMINAL statements:

Set RECOVERYLEVEL to HIGH or INTERMEDIATE or NONE (the default).

Note The recovery level for an ACB is set when the ACB is opened. Subsequent use of the ACB will use the same recovery level as set when the ACB was opened. By default ACBs remain open until Session Manager terminates or the ACB fails. To close an ACB when no sessions are active on that ACB, on the SYSTEM statement set the CLOSEACBINACT to YES.

SYSTEM statement SESACB considerations

Sessions started with the SESACB session must either associate the session with an APPL statement which has the RECOVERYLEVEL specified or use RECOVERYLEVEL on the USER statement.

ACBs and RANGE statements considerations

Session Manager nodes within a High Level Availability Sysplex environment must not use the same ACBs. It is recommended to share the configuration across nodes in a Sysplex environment but if ACBs are specified on the shared RANGE statements then they must be unique.

One possible set-up to overcome this situation would be to add to the SYSTEM statement:

```
%% let gc_acb_suffix = 'cc'
```

(where cc is unique for each SYSTEM statement) and to change your RANGE statements to include this variable, for example:

```
RANGE CICSH
FROM ISZ%%gc_acb_suffix%%001
TO fff
HEX
```

VTAM MNPS settings

(Only required for sessions with RECOVERYLEVEL set to HIGH.)

For an ACB to be eligible to support MNPS it must be defined to VTAM with the PERSIST=MULT parameter on the VTAM APPL statement. In order to support VTAM MNPS the Session Manager Controller, the Standby Controller and the target applications must use the z/OS VTAM High Performance Routine option set to Rapid Transport Protocol (HPR=RTP) paths. See the *SNA Network Implementation Guide* for details of the MNPS network requirements.

How to set the Sysplex audit log

On the SYSTEM statement on the SYSPLEXGROUP parameter for each instance or in the common System statement:

Set LOGSTREAMNAME to the z/OS system logger log stream defined by your installation, to capture audit messages across a Sysplex group. If the LOGSTREAMNAME parameter is not specified then Session Manager will not store audit messages on a common Sysplex log that can be viewed by any instance in the group.

Other SYSPLEXGROUP parameters

Please review the SYSPLEXGROUP parameter on the SYSTEM statement in the *Technical Reference* manual and set the additional parameters as required.

Sample signon validation exit script (E21)

The E21 exit point is invoked each time a user attempts to sign on to Session Manager (for details, see ‘Signon validation exit point (E21)’ on page 217). In a Parallel Sysplex environment, if a user is already signed on to (or disconnected from) a particular Session Manager node then if the user attempts to sign on again the sample E21 exit script, ISZE21GA, provides a mechanism to transfer the signon to the original Session Manager node. If using the sample exit script ISZE21PH then the same mechanism is provided via use of a CF user structure. For a description of this, see ‘Automatic user reconnection within a Sysplex’ on page 389.

Notes

- 1 This mechanism does not require the use of Parallel Sysplex Generic Resources; it can operate in any VTAM network where multiple instances of Session Manager are connected using Session Manager networking.
- 2 The mechanism utilizes the VTAM CLSDST PASS facility and so requires the terminal to be connected by VTAM to Session Manager. However, the mechanism can still operate if the terminal is connected via TCP/IP to, for example, IBM’s Communications Server, which then connects to Session Manager via a pool of VTAM ACBs.

When the sample E21 exit script is invoked it will:

- Check all connected Session Manager nodes to see if the user attempting to sign on to a particular node has already signed on to another node.
- If the user is already signed on to (or disconnected from) another Session Manager node then it will transfer the signon to that node after calling the E21 Assembler exit to perform authentication.

How to implement the sample E21 exit script

To implement the sample E21 exit script, complete these steps:

- 1 Use the LOCALNODE parameter of the SYSTEM configuration statement (see the *Technical Reference*) to specify a unique node name for each Session Manager system in the group.

For example, for ‘System A’ use a SYSTEM statement like this:

```
SYSTEM
...
LOCALNODE SYSA
...
```

- 2 Use LINK configuration statements (see the *Technical Reference*) to connect the Session Manager systems in the group.

See also:

- ‘Session Manager networking’ on page 373
- and
- (if using OLA) the section on multiple instances of Session Manager sharing a configuration in the *Online and Batch Administration* manual.

- 3 Use RUSER configuration statements (see the *Technical Reference*) to specify a 'default' remote authority so that users can issue these commands to remote Session Manager systems in the group:
 - QUSER
Can be used to establish the status of a particular userid (for details, see the *Technical Reference*).
 - QUSERCMD
A sample command script which issues the QUSER command to query a userid from a remote node.

For example, if there are three remote Session Manager systems having node names of SYSA, SYSB and SYSC respectively, use these RUSER statements:

```
RUSER * NODE SYSA AUTH 1
RUSER * NODE SYSB AUTH 1
RUSER * NODE SYSC AUTH 1
```

- 4 Change the node name entries contained in the sample E21 exit script (these entries are of the form `let lc_e21node.n = 'node'`) to reflect the node names specified using the LOCALNODE parameter (see step 1 above).
- 5 Make the sample E21 exit script and the QUSERCMD command script available on all Session Manager nodes in the group.
- 6 Use the OPTION configuration statement (see the *Technical Reference*) to specify the use of the sample E21 exit script on all nodes:

```
OPTION EXIT ISZEXT00 E21 S
```
- 7 Use the GENERICACB parameter of the SYSTEM configuration statement (see the *Technical Reference*) to specify a VTAM common (or generic) ACB name in the Parallel Sysplex environment.

Summary

In a Parallel Sysplex environment, Session Manager provides facilities to support:

- Workload balancing
- Sharing of the Session Manager configuration
- Sysplex networking
- Automatic user reconnection.
- Sysplex Group wide BROADCAST and MSG commands
- Sysplex Group audit log.
- VTAM application session recovery.

If a user is already signed on to (or disconnected from) a particular Session Manager node then if the user attempts to sign on again a sample E21 exit script provides a mechanism to transfer the signon to the original Session Manager node.

Using Session Manager as a 'front-end'

Session Manager can be configured so that the facility of running multiple sessions, with the ability to transfer control between them using escape command sequences, is suppressed.

When this is done, as soon as a session is activated from the Menu panel, Session Manager relinquishes control of the physical terminal, and transfers it completely to the application the user has selected.

In effect, Session Manager is acting only as a 'front-end' to the applications a user might select. The facility provides the means of presenting a secure system, if the Session Manager system has been configured to display the Signon panel, and a user friendly Menu display at the terminal being used.

The subjects covered in this chapter are:

- 'Overview' on page 426
- 'Impact on terminal definition and logon' on page 427
- 'Using a selective signon to Session Manager' on page 428
- 'Logmode table entry name implications' on page 430
- 'Summary' on page 431

Overview

This facility is provided by the CLOSELOGOFF and CLOSEDISC subparameters of the SESSION parameter of the PROFILE, USER and TERMINAL statements, or by the APPL statement. Both cause Session Manager to relinquish control of the terminal to the application when the selected key is pressed and as far as the user is concerned, Session Manager 'disappears' temporarily. There is an important difference between the two subparameters:

- **CLOSELOGOFF**
Causes any other active sessions to be terminated when that session is selected. Therefore, this subparameter provides a means of forcing a user to have only a single active session at any time.
- **CLOSEDISC**
Causes Session Manager to issue a DISCONNECT for the terminal, so any other sessions that were active remain so.

On exiting from the selected application, control of the terminal can be returned to Session Manager provided that the LOGAPPL parameter of the LU statement defining the terminal to VTAM[®] is specified. For more information, see 'Impact on terminal definition and logon' on page 427.

Note If SHAREDISC or SHARESESS are being used to implement the Shared User Facility (see 'The Shared Userid facility' on page 319) then CLOSELOGOFF will behave as CLOSEDISC for the primary user (the first terminal logged on with the userid), but for secondary users CLOSELOGOFF will behave as usual.

Impact on terminal definition and logon

What happens to the terminal display when a user quits from a session activated using either CLOSELOGOFF or CLOSEDISC depends on the way in which the terminal is defined to VTAM and to Session Manager.

The terminal is defined to VTAM by an LU statement and the LOGAPPL parameter of this statement may be used to cause the terminal to be automatically logged on to a specific application. This therefore enables the terminal to be automatically logged back on to Session Manager when a user quits from an active session. If the LOGAPPL parameter is not specified, the terminal is returned to VTAM control and usually a VTAM logon screen would be displayed. In order to access another application, the user has to log back on to Session Manager again.

Assuming that the terminal has been defined with a LOGAPPL of ISZSMGR, either the Session Manager Signon panel or the appropriate Menu panel is displayed, depending on how the terminal is defined to Session Manager. Normally the Signon panel would be displayed, unless SIGNON NO has been specified on the SYSTEM statement, or an applicable TERMINAL statement has been defined. All of these parameters are described in the appropriate sections of 'System definition' in the *Technical Reference* manual.

Therefore, to avoid the user having to log back on to Session Manager each time the active session is terminated:

- 1 Specify LOGAPPL ISZSMGR on the LU statement
- 2 Define an appropriate TERMINAL statement with SIGNON NO or specify SIGNON NO on the SYSTEM statement.

Using a selective signon to Session Manager

When Session Manager is being used as a front-end, as described in 'Overview' on page 426, it is possible to re-access Session Manager and force a signon, even though a signon was not used for the first access.

There are situations in which this facility can be particularly useful. Suppose an Installation requires the Session Manager Menu panel to be displayed at all terminals, with no preceding Signon panel, but for some applications, users are required to enter a signon. This might be the case when most applications are generally available, but certain transactions are only available to users with a high security clearance.

The Session Manager control statements need to be appropriately configured to achieve this, and the following example shows how it might be done:

```

SYSTEM    ACB    ISZSMGR
          SIGNON No

TERMINAL  Term1
          SIGNON Yes
          SIGNONPANEL Screen1

PROFILE   Normprof
          ...
          SESSION 24
          APPLID  ISZSMGR
          DATA  '////Term1'
          CLOSEDISC

USER      Secure
          PASSWORD person
          PROF  Sctyprof
  
```

The SYSTEM statement has SIGNON NO specified, which means that no Signon screen is displayed when Session Manager is first accessed. When session 24 is selected, a signon to Session Manager is issued – remember that specifying CLOSEDISC causes the first level Session Manager to relinquish control at this point. This is sometimes known as a recursive signon. The APPLID parameter for the session must specify the same name as the SYSTEM statement ACB parameter, in this instance ISZSMGR.

The DATA subparameter for session 24 specifies Term1 as the terminal definition to be used for this key. Since the TERMINAL statement for Term1 defines a Signon panel of Screen1, this will be displayed when the session is selected. Therefore, provided that the user has a signon code, (shown in the example as Secure Person), the Menu panel defined by the profile Sctyprof will then be displayed at the terminal.

Thus, the requirement for all users to be able to access the majority of applications without the need to sign on to the system, while at the same time forcing a signon for a subset of security sensitive applications is satisfied.

To return to the initial Menu panel, the command LOGOFF SIGNON should be issued. Since the default profile has the CLOSEDISC option and the SYSTEM statement has been specified with SIGNON NO, the original Menu panel is displayed. Alternatively, a normal LOGOFF may be issued, provided that the second level profile Sctyprof has LOGDISC SIGNON defined. This circuitous procedure is necessary because VTAM does not permit an automatic re-logout to an application from which a user has just logged off.

The common user parameter SIMRECON may be used to modify the need for a user to sign-on again after a CLOSEDISC. If SIMRECON Yes is active when the CLOSEDISC session terminates, Session Manager automatically reconnects and displays the user's menu without requiring the user to signon again using the signon panel. Further information on the SIMRECON parameter may be found in the *Technical Reference* manual.

Logmode table entry name implications

When Session Manager passes control to an application and the session has CLOSEDISC or CLOSELOGOFF specified, it passes across any data defined using the DATA parameter and also the logmode table entry name which it has determined should be used for the session. A full description of the way in which logmode names are selected can be found in 'Application session logmode entry selection' on page 185.

Because of restrictions in VTAM, it is necessary that this logmode entry name appears either in the logmode table assigned to the terminal or in the IBM default logmode table. Failure to do this can cause message 273E to be issued and the session may not be established as expected.

Summary

The following commands and parameters are relevant to this facility; appropriate PROFILE and TERMINAL statements should also be defined, if required.

Command or parameter	Description
CLOSELOGOFF subparameter	Specifies that other active sessions are to be terminated when the specified session is selected. May be entered at session level on the SYSTEM, PROFILE, USER, APPL and TERMINAL statements.
CLOSEDISC subparameter	Specifies that other active sessions are to be disconnected, but remain active, when the specified session is selected. May be entered at session level on the SYSTEM, PROFILE, USER, APPL and TERMINAL statements.
LOGAPPL parameter	Should be used to define a logappl of ISZSMGR, on the LU statement in the VTAM definitions. When the session is terminated, control is passed to Session Manager. What is shown depends on which of the two subparameters above were specified and whether SIGNON YES or NO has been specified.
SIGNON parameter	Specifies whether the Signon panel is displayed when Session Manager is accessed. May be entered at session level on the SYSTEM and TERMINAL statements. See the examples in 'Using a selective signon to Session Manager' on page 428.
DATA parameter	Specifies the data that is to be passed to the session.
LOGOFF command	This should be specified as LOGOFF SIGNON when logging off from Session Manager, unless LOGDISC SIGNON has been specified on a secondary level profile. See the example in 'Using a selective signon to Session Manager' on page 428.

The Application Builder feature

This feature enables a new application to be 'built' by combining the output from one or more applications. Output is gathered selectively from two or more sessions and presented together on the screen. The information gathered may be presented in any desired layout on the screen by using a panel definition. In addition, any chosen output from a session can be used as script input to another session.

The subjects covered in this chapter are:

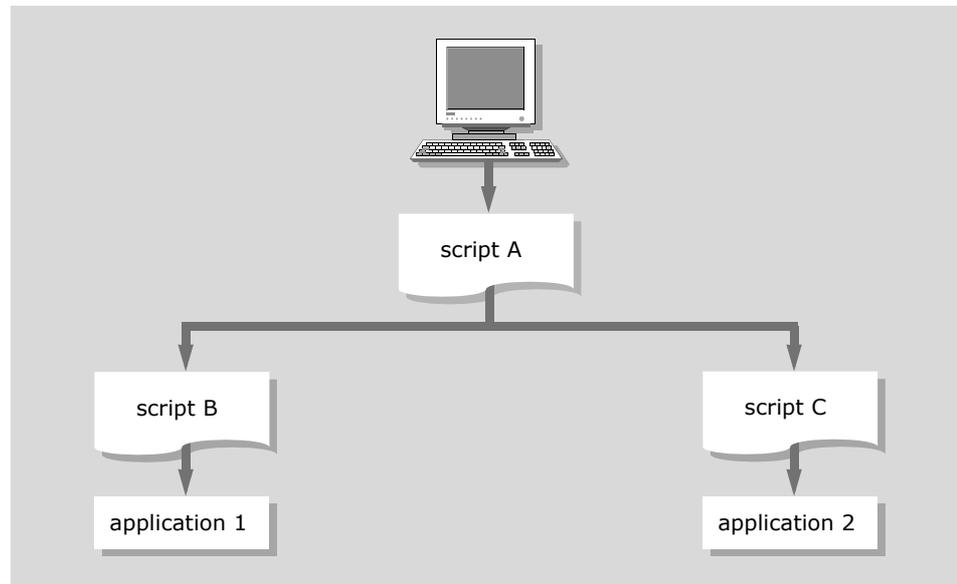
- 'Overview' on page 434
- 'Application Builder and remote sessions' on page 435

Overview

A new application is built by running scripts on each of the sessions concerned. As with normal Input scripts, each can provide input to its application, await output, and scan the output for various content. However, using the Application Builder verbs, each script can also:

- start a session
- terminate any specified session
- run a specified script on any particular session
- halt a script in progress on any particular session
- copy any part of its application screen to a variable
- send variable data to a script running on another session
- receive variable data from a script running on another session

An illustration of how the new application can be ‘built’ is provided below.

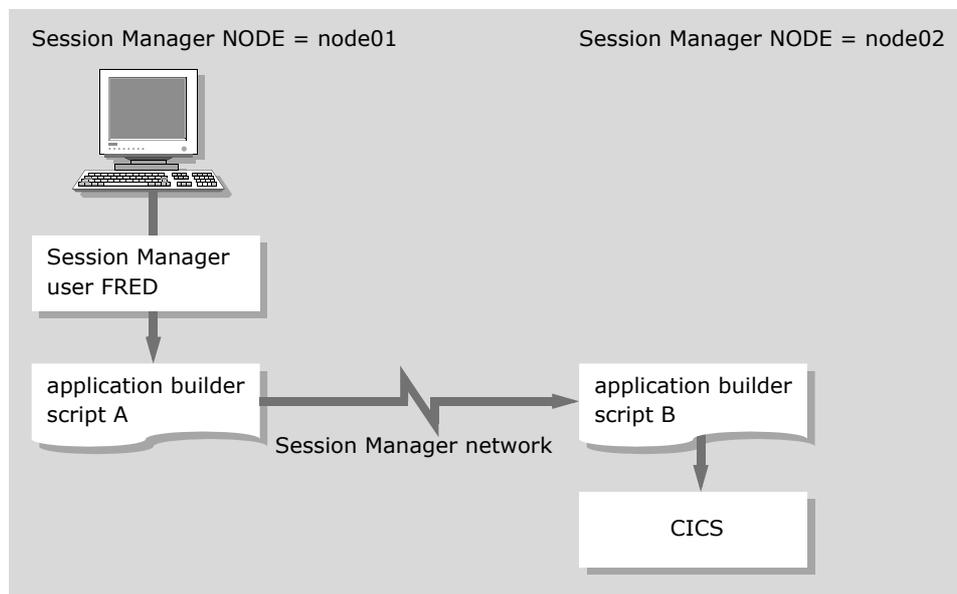


Script B and script C act like normal session scripts, but in addition they send data, which could be part of the application screen, to script A. Script A displays a panel on the screen, in which various fields have been defined to contain the data received from scripts B and C. Script A can send any data that is input at the terminal to script B, script C, or both. It can also send data received from script B to script C and data received from script C to script B.

Application Builder and remote sessions

If the Session Manager networking feature is enabled, Application Builder can also be used when one of the sessions involved is a remote session. The RPARTNER operand of the SENDDATA application builder verb enables communication between a local application builder script and its remote partner.

For example, in the following diagram, the user FRED has signed on to a terminal at node01 and started a remote session in node02. Script A has been specified as the local startscript by defining it in the local definitions and script B has been specified as a remote startscript by defining it in the appropriate remote APPL definition. By specifying the RPARTNER operand on the SENDDATA parameter in place of the SESSION operand, script A can send data through the network to script B. In exactly the same manner, script B can send data through the network to script A.



Full details of the Application Builder script verbs are given in the *Panels, Scripts and Variables* manual.

Monitoring performance statistics

IBM Session Manager for z/OS provides two types of performance monitoring routine. The first is a monitoring routine to gather performance data on itself. The second provides session level statistics on network and application response times.

The subjects covered in this chapter are:

- 'Performance Monitor' on page 438
- 'Response Time Monitor' on page 440

Performance Monitor

Session Manager contains a monitoring routine to gather performance data about itself. The routine is part of the User Exit module and is invoked at exit point E08 every ten minutes. For details of these and other changes required to implement the routine refer to 'Implementing the Performance Monitoring routine' on page 439.

The routine collects data for ten-minute intervals and each interval produces a record. The length of an interval may vary on a low-activity system because the exit may not be invoked. The records are held in storage in a wrap-around table containing 100 records. The exit could be amended to write the records to a file. For example, SMF records could be written.

The Performance Statistics panel

To view the performance data, enter the command:

```
PERFPAN
```

from the Menu panel. The panel can be paged forwards and backwards and, because the records are wrapped, the latest record is always highlighted. Data does not appear during the first 20 minutes after Session Manager initiation, but from that point on, it is produced at 10 minute intervals. The panel is not automatically refreshed. Pressing the Enter key refreshes the panel.

Session Manager		Performance Statistics						dd/mm/yyyy hh:mm:ss		
LU	luname							user		
Date	Time	Elapsed secs	Terminal in	Terminal out	Session in	Session out	TCB secs	SRB secs	Total secs	CPU rate
20/11/09	16.45.52	806.19	.33	.52	.30	.49	23.62	7.02	30.64	3.80
20/11/09	17.04.02	1091.21	.18	.21	.15	.18	20.09	6.02	26.11	2.39
20/11/09	17.16.45	762.06	.23	.25	.19	.21	20.09	4.29	24.38	3.19
20/11/09	17.20.24	219.46	.36	.42	.32	.37	7.85	1.65	9.50	4.32

====>

PF1:Help PF3:Quit PF4:Return PF7:Bwd PF8:Fwd

On the left is the date and time that each record has been created. The elapsed time of the interval is shown in seconds. On a system where the exit is regularly invoked, each interval will be approximately 600 seconds.

The average number of inputs and outputs per second for that interval to all the terminals in the system is shown next, followed by the average number of inputs and outputs per second to all the active sessions. The session outputs figure includes non-visible outputs, even those that are discarded.

The TCB secs value reflects the amount of CPU used by Session Manager. The SRB secs value shows the amount of CPU used for system requests on behalf of Session Manager. This figure is in effect the overhead of running Session Manager. The sum of these two figures are added together to give the total CPU seconds used in the interval.

The CPU rate is calculated as follows:

$$(\text{total secs/elapsed seconds}) * 100$$

In a multi-processor environment, Session Manager must be run with `AFFINITY` set for one processor. This means that the CPU rate shown will be the effective percentage for that processor only.

Implementing the Performance Monitoring routine

To implement the routine a user exit must be active in Session Manager. The source is supplied with a member name of `ISZE08CP`. If this exit is to be used with the Multiple Exit Driver, it should have an executable module name of `ISZEXT08`.

Ensure that source member `PERFPAN` is included in the system. This is the supplied panel definition for the Performance data panel. It can be loaded dynamically using the `UPDATE` command.

The final change that needs to be made in each operating system is to amend each Menu panel from which the `PERFPAN` command might be invoked. The following TPSL statements should be included in the `PROCESS` section:

```
If t_command = 'perfpan'      /* if perfpan entered
    Let uc_oldpan = t_panel    /*   save current panel
    Let t_panel = 'perfpan'    /*   set new panel
    Let t_command = ' '        /*   clear command line
End
```

The amended menu panels should then be included in the Session Manager system.

Response Time Monitor

The Response Time Monitor provides session level statistics on application and network response times. This response is the length of time the system takes to respond to terminal input. This consists of two parts: the network time and the application time.

Session Manager can measure the application response time accurately. However, for the network response time, a session manager cannot detect the time a user takes to press Enter at a terminal, nor can it detect the time the output is displayed on the terminal by the system. The session manager does know when it sends the output to the system and the time the response to the output is received from the system. The difference between these two times can be taken as the equivalent of a terminal's network response time. The session manager processing time is included in the network response time, although the value will be negligible. See 'Notes on response time monitoring' on page 442 for further points to consider.

If a session is not the current display when output arrives from the application, the application count is incremented, but the total and network counts remain the same. This is because the outputs are not in response to a terminal input. Similarly, when a script is inputting data, application outputs are not being sent to the terminal so the same applies.

Data is available on all sessions with the STATS parameter active and this data can be accessed by the E39 Session Termination exit and also by panels, scripts, and other exits via the s_rtm variables.

The Monitor maintains three sets of responses

- Network Responses
- Application Responses
- Total Responses (which are Network responses + Application responses).

For each of these sets of responses, the following are maintained

- Average response time
- Five latest response times
- Five longest responses, together with the date and time they occurred.
- Three counts for fast, medium, and slow responses. These categories are defined by thresholds and these control which count to use.

All the response times and percentages are subject to rounding.

The following parameters of the SYSTEM statement are used to define the Threshold values:

- RTMT1 nnnnn
- RTMT2 nnnnn

where nnnnn is the threshold values in milli-seconds. The value specified must be in the range 0-32767 and the RTMT1 value should be less than that specified on RTMT2. The defaults are RTMT1=1000 (1 second) and RTMT2=5000 (5 seconds).

Therefore, response times less than, or equal to, the value set in RTMT1 are considered 'fast', those responses in the range between the RTMT1 and RTMT2 values are considered 'medium' and those that exceed the RTMT2 value are considered 'slow'.

There is no internal checking of the values you specify for these parameters, but if the RTMT1 value is specified as greater than the RTMT2 value, or is zero, then the statistics produced will have little benefit. If zero is specified for one value, there will only be two thresholds used for statistics production.

If these values are updated, existing sessions continue to use the old values until a QUERY command is issued with RTM and RESET parameters.

The following QUERY commands:

```
QUERY USER | LU | TERMINAL | TN3270
```

can be used with the parameters:

```
RTM [RESET] | RTMALL [RESET]
```

to display or reset response time statistics.

where the parameters have the following meanings:

RTM	Displays the total response time counts per session.
RTMALL	Displays all counts, that is, application, network, total response times per session.
RESET	Causes all counts to be reset

Examples of response time statistics

Example 1

If RTMT1 is set as 200 and RTMT2 at 300, the command:

```
Q USER JOHN RTM
```

outputs the following:

```
USER JOHN PROFILE PROF1 LU S05TVV01
(2) S10VM ACB S10JC009 LGMD D4B32795 status -A- S( ),P( ) Current
Session started at 11/20/02 13:34:19
Average Response = 0.60 seconds
Not exceeding 0.20 = 7 ( 70%)
Exceeding 0.20 but not exceeding 0.30 = 1 ( 10%)
Exceeding 0.30 = 2 ( 20%)
Latest 5 = 0.15, 0.15, 0.15, 0.15, 0.24
Longest = 2.53 (11/20/02 13:38:41)
2nd Longest response = 2.27 (11/20/02 13:38:39)
3rd Longest response = 0.24 (11/20/02 13:38:44)
4th Longest response = 0.15 (11/20/02 13:38:58)
5th Longest response = 0.15 (11/20/02 13:38:47)
```

Example 2

With the same settings, the command:

```
Q USER JOHN RTMALL
```

outputs the following:

```
USER JOHN PROFILE PROF1 LU S05TVV01
(2) S10VM ACB S10JC009 LGMD D4B32795 status -A- S( ),P( ) Current
Session started at 11/20/02 13:34:19
Total: Average Response = 0.60 seconds
Total: Not exceeding 0.20 = 7 ( 70%)
Total: Exceeding 0.20 but not exceeding 0.30 = 1 ( 10%)
Total: Exceeding 0.30 = 2 ( 20%)
Total: Latest 5 = 0.15, 0.15, 0.15, 0.15, 0.24
Total: Longest = 2.53 (11/20/02 13:38:41)
Total: 2nd Longest = 2.27 (11/20/02 13:38:39)
Total: 3rd Longest = 0.24 (11/20/02 13:38:44)
Total: 4th Longest = 0.15 (11/20/02 13:38:58)
Total: 5th Longest = 0.15 (11/20/02 13:38:47)
Network: Average Response = 0.33 seconds
Network: Not exceeding 0.20 = 9 ( 90%)
Network: Exceeding 0.20 but not exceeding 0.30 = 0 ( 0%)
Network: Exceeding 0.30 = 1 ( 10%)
Network: Latest 5 = 0.09, 0.09, 0.08, 0.09, 0.17
Network: Longest = 2.47 (11/20/02 13:38:41)
Network: 2nd Longest = 0.17 (11/20/02 13:38:44)
Network: 3rd Longest = 0.10 (11/20/02 13:38:39)
Network: 4th Longest = 0.09 (11/20/02 13:38:58)
Network: 5th Longest = 0.09 (11/20/02 13:38:51)
Application: Average Response = 0.27 seconds
Application: Not exceeding 0.20 = 9 ( 90%)
Application: Exceeding 0.20 but not exceeding 0.30 = 0 ( 0%)
Application: Exceeding 0.30 = 1 ( 10%)
Application: Latest 5 = 0.06, 0.05, 0.06, 0.06, 0.06
Application: Longest = 2.17 (11/20/02 13:38:39)
Application: 2nd Longest = 0.06 (11/20/02 13:38:44)
Application: 3rd Longest = 0.06 (11/20/02 13:38:47)
Application: 4th Longest = 0.06 (11/20/02 13:38:58)
Application: 5th Longest = 0.06 (11/20/02 13:38:50)
```

Example 3

The command:

```
Q USER JOHN RTMALL RESET
```

outputs as above followed by the line:

```
Session response counts reset
```

Subsequent RTM or RTMALL queries output the additional line:

```
Session response counts reset at 11/20/02 13:38:18 by OPERATOR CONSOLE
```

Notes on response time monitoring

A delay in the delivery of a terminal input to the session manager cannot be identified unless this delay also occurs whilst sending terminal output. For example, the network is unavailable for 10 minutes when the user presses Enter but when the network delivers the input to the session manager, the system responds normally. In this case, the response recorded would be a normal response but the user's real response was greater than 10 minutes.

In networks involving intermediate components which reply with responses before the output has reached the terminal, the recorded network response time will be a portion of the total response time.

**Usage with
TCP/IP**

For TN3270 terminals, Session Manager internally handles outputs and responds immediately on sending the request to the TCP/IP network. TCP/IP has no mechanism to respond. This means the response time times on the terminal side for TCP/IP will be incorrect. The application response times will be accurate for all sessions.

Extracting response time statistics

Session variables are available that can be used by the E39 Session Termination exit to extract these response time statistics. These session variables are described in the section titled 'Response time monitor session variables' in the 'Session Manager variables' chapter of the *Panels, Scripts and Variables* manual.

CHAPTER 20

Converting a Classic to an OLA system

This chapter explains how to convert from a Classic to an Online and Batch Administration (OLA) system.

The subjects covered in this chapter are:

- ‘Overview’ on page 446
- ‘Conversion procedure’ on page 448
- ‘The OLA Enabler’ on page 451
 - ‘Installing an OLA system’ on page 451
 - ‘Running the OLA Enabler’ on page 451
- ‘Setting up OLA for single and multiples Session Manager instances’ on page 460
- ‘Creating the Session Manager started task’ on page 460

For details of the post-installation OLA configuration steps, please see the *Online and Batch Administration* manual.

Overview

Datasets used by Session Manager

The configuration is stored in one of two possible formats, depending on whether or not you will be using Online and/or Batch Administration to tailor the product.

If Online and/or Batch Administration is not used – Classic

If Online and/or Batch Administration is not used to tailor the product, control statements are read from a Classic configuration which is created as part of the installation process. In a Classic configuration, all configuration definitions are stored in members of PDS(s) allocated to the DDNAME of CONFIG.

If Online and/or Batch Administration is used – OLA

If Online and/or Batch Administration is used to tailor the product, configuration data is stored in several PDS(E)s. In this configuration, each PDS(E) is allocated to a particular DDNAME and *must be maintained exclusively by Online and/or Batch Administration*.

Summary of conversion procedure

OLA allows you to tailor Session Manager using a series of menus, lists and attribute display panels. For more information on OLA, refer to the *Online and Batch Administration* manual.

If your existing configuration is Classic (that is, all configuration definitions are stored in members of PDS(s) allocated to the DDNAME of CONFIG), then you must complete the following steps to implement the new format configuration. The steps are described in more detail in ‘Conversion procedure’ on page 448.

- 1 Apply a default, system-wide OLA security class. To do so, add the parameter OLAClass US to the SYSTEM statement in your ISZCONxx configuration member.

For more information on the OLAClass parameter, refer to the *Online and Batch Administration* manual.

- 2 Complete these steps, which are described in detail in ‘The OLA Enabler’ on page 451:
 - a Install an OLA system. See ‘Performing a basic Classic or OLA installation’ on page 57.
 - b Run the OLA Enabler and check that it completes successfully. A sample job to do this can be found in member ISZOEJOB (in library .SISZCONF). If necessary, the ISZOEJOB will also copy any modified or added panels from your Classic system to your OLA system.
 - Edit the sample configuration ISZCONOE and change COPY ISZCON01 to COPY ISZCONxx, where xx is the unique identifier for your current configuration. Your ISZCONxx member must include the COPY ISZOLA and COPY ISZCLSO statements as specified in the supplied ISZCON01 member.

- If necessary, you will also need to add `COPY xxxxxxxx` statements to your `ISZCONxx` member (in library `.SISZCCNF`) for each modified or added panel.
- c** Add Language Packs (LPs):
- The default LP and any additional Language Packs are generated by the installation of the OLA system (see step a above).
 - The default LP is defined in the `COPY ISZLEN2` statement in the supplied `ISZCON02` member, where `EN` is the language id for English.
 - To load a new LP, add an additional `COPY ISZLyy2` statement, where `yy` is the 2-character language id for the new language, to your `ISZCON02` member.
- 3** A sample procedure for the new OLA-enabled Session Manager started task can be found in member `ISZCOLA` (in library `.SISZCONF`). Tailor the `JCL` in this sample procedure to your Installation's requirements.
- 4** In your test environment, start the tailored `ISZCOLA` started task. Check that the OLA-enabled Session Manager system starts up successfully and that *the configuration is error free*.
- 5** Test the new OLA-enabled Session Manager system to ensure that the product has been correctly configured.
- 6** Shut down the new OLA-enabled Session Manager system when you have verified that everything is working as expected.
- 7** Complete the following steps to put the new release into production:
- a** Using OLA, change the `ACB ISZSMGR` and `SESACB ISZ001` parameters on the `SYSTEM` statement to match the `VTAM` `acbnames` in your production environment. (Reply **N** when the `Immediate activation of change?` prompt is displayed.)
 - b** To allow you to use your applications, review the `FROM` parameter on any `ACB RANGE` statements and, using OLA, change the values to correspond to the `VTAM` `acbnames` in your production environment. (Reply **N** when the `Immediate activation of change?` prompt is displayed.)
 - c** At a convenient time, shut down your existing production Session Manager system and replace the procedure that starts this system with the procedure that starts the new OLA-enabled Session Manager system.
 - d** Re-start the production Session Manager system.

Conversion procedure

If you have decided to run Session Manager in OLA mode you need to carry out various reconfiguration tasks.

How reconfiguration is achieved

In an OLA-enabled configuration, configuration data is stored in several PDS(E)s and a specific DDNAME is associated with each statement type (APPL, PROFILE, USER, and so on).

A configuration member, ISZCONxx, is required for each Session Manager instance that shares the configuration, where xx is the unique identifier for the instance. A value of BT is not allowed as this is used by Batch Administration.

Each ISZCONxx member, which is stored in the PDS(E) allocated to the DDNAME of CONFIG, is created to contain the following definitions; for details, see 'ISZCONxx definitions' on page 143:

```
OPTION parms
COPY ISZCINIT
INSTALLSU
AUDITROUTE parms
TRACEROUTE parms
%% let gc_acb_prefix = 'acb_prefix'
%% let gc_acb_suffix = 'acb_suffix'
COPY ISZCOMON
```

Member ISZCOMON

The ISZCOMON member, which is processed at Session Manager start up, is created to contain a number of PCOPY statements, each of which loads all members of the PDS(E) allocated to a specified DDNAME (see 'Configuration DDNAMEs' on page 448), and COPY statements for system panels and scripts.

Note OLA recognizes only DDNAMEs that are associated with the configuration. Therefore, do *not* add PCOPY statements to member ISZCOMON (or any other member) for DDNAMEs other than those associated with the configuration.

Configuration DDNAMEs

To use Session Manager in OLA mode, the DDNAMEs listed below are required. Except for CONFIG, each DDNAME is associated with a particular statement type; the PDS(E)s associated with the DDNAMEs (see 'Customer, samples and system configuration datasets' on page 155) *must be maintained exclusively by Online and/or Batch Administration.*

DDNAME	Configuration statement	Contents
ACTIVE		Control information for ISZACTV Batch command
APPL	APPL	Application definitions.
ASCRIP	SCRIPT	Authorized (system) script definitions. See also Notes 1 and 3 below.
CLASS	OLAClass (see Note 2)	<i>(For internal Session Manager use only.)</i> Security class definitions.
COMMAND	COMMAND	Command authority definitions.
CONFIG		An ISZCONxx member for each Session Manager instance sharing the configuration. See also: <ul style="list-style-type: none"> ▪ Notes 3 and 4 below. ▪ ‘Supplied system CONFIG members’ on page 450.
GROUP	GROUP	Group definitions.
HCFORMAT	HCFORMAT	Hardcopy format definitions.
HCPRF	HCPROFILE	Hardcopy profile definitions.
HCRUTE	HCRUTE	Hardcopy route definitions.
LINK	LINK	Link definitions.
MASK		Masks for ISZSRCH Batch command
MESSAGE	MESSAGE	Message definitions.
NODE		Sysplex node.
PROFILE	PROFILE	Profile definitions.
RANGE	RANGE	ACB range definitions.
RUSER	RUSER	Remote user definitions.
SEARCH		Control information for ISZSRCH Batch command
SYSTEM	SYSTEM	A ISZSYSxx system definition member for each Session Manager instance sharing the configuration and the common system definition ISZSYSCM.
TEMPUSER	USER	<i>(For internal Session Manager use only.)</i> Temporary user definitions.
TERMINAL	TERMINAL	Terminal definitions.
TRANSTAB	TRANSTABLE	Translate table definitions. See also Note 3 below.
UPDATE		Control information for ISZUPDT Batch command
USCRIP	SCRIPT	Unauthorized (user) script definitions. See also Notes 1 and 3 below.
USER	USER	User definitions.
VTM		Virtual terminal masks.

Notes

- 1 All OLA scripts are *authorized*, which means that they are able to issue product commands with a Session Manager user authority (AUTH) of 9, the highest user authority. (For more information on authorized scripts, please refer to the *Panels, Scripts and Variables* manual.)

To update the configuration, OLA uses this facility to invoke the Session Manager PUPDATE command on behalf of a standard user. Typically, a standard user would have an AUTH of just 1 which is, by default, lower than the AUTH required to invoke the PUPDATE command.

- 2 OLAClass is a common enduser configuration parameter.
- 3 Each ISZCONxx configuration member, panel, script or TRANSTABLE should be maintained (using the ISPF editor) by the Session Manager implementor; all other configuration datasets *must be maintained exclusively by Online and/or Batch Administration*. See also 'ISZCONxx definitions' on page 143.
- 4 AUDITROUTE and TRACERoute definitions are stored in the ISZCONxx configuration member(s) – unlike the other configuration statements, which are associated with a particular DDNAME/PDS(E). Full descriptions of all Session Manager configuration statements can be found in the *Technical Reference*.

**Supplied system
CONFIG
members**

Several members are supplied in the system CONFIG dataset, which *should not be deleted or renamed*:

Member	Description
ISZCOMON	Contains a PCOPY statement for the DDNAMEs listed in 'Configuration DDNAMEs' on page 448 and COPY statements for system panels and scripts.
ISZCINIT	Contains Session Manager maintenance – that is, PATCH or PATCHSU definitions. See also Note 1 below.
ISZCSnnn	Each member contains maintenance specific to a particular SU. See also Notes 1 and 2 below.
ISZCPnnn	Each member contains messages and, potentially, other configuration definitions specific to a particular SU. These messages and configuration definitions are used by the code for the particular SU during the load of the SU.
ISZRSVRD	<i>(For internal Session Manager use only.)</i> Used by PUPDATE command processing.

Notes

- 1 Do not store PATCHes supplied by IBM here unless explicitly told to do so by IBM.
- 2 Contains PATCH definitions that must be applied during the load of the SU by using the INSTALLSU statement in the ISZCONxx members.

For information on customer versus system datasets in an OLA system see 'Customer, samples and system configuration datasets' on page 155.

The OLA Enabler

After you have confirmed that *your configuration is error free*, to use OLA to complete these tasks:

- 1 Install an OLA system. See ‘Performing a basic Classic or OLA installation’ on page 57.
- 2 Run the OLA Enabler to implement the new configuration. The OLA-enabled configuration is described in ‘Conversion procedure’ on page 448.

Installing an OLA system

See ‘Performing a basic Classic or OLA installation’ on page 57.

Running the OLA Enabler

After you have successfully installed the basic OLA system, run the OLA Enabler to implement the new format configuration.

- The OLA Enabler command can be found in member ISZOLAEN, which is supplied in library .SISZCONF.
- A sample job to run the ISZOLAEN command can be found in member ISZOEJOB. This job uses sample configuration member ISZCONOE. Both components are supplied in library .SISZCONF.

To run the OLA Enabler, complete these steps; if any problems are encountered, please contact your local Support Representative:

- 1 Edit the sample job ISZOEJOB and make these changes:
 - a Change the DSNPREF= parameter to match the prefix that you specified when you installed the OLA system and a new set of configuration datasets was allocated.
 - b Change the LOADDSN= parameter to reference the supplied LOADLIB.
 - c Tailor the ISZOLAEN options, which appear at the bottom of ISZOEJOB, to your Installation’s requirements (see ‘OLA Enabler parameters’ on page 453). In particular, change the identifier for the Session Manager instance from 01 to xx, where xx is the unique identifier for your current configuration.
 - d Review the values of parameters batlnk, onl1nk and com1nk (see ‘Example’ on page 456) and the nolinks parameter to establish if the generated OLA Session Manager will run in a Sysplex-only environment or in a non-Sysplex environment or in a mixed environment.

Also review ‘Networking and Sysplex considerations (if required)’ on page 127 and ‘Parallel Sysplex support’ on page 387.

- e If you have not modified or added any panels to your Classic system then Step 2 should be removed from the ISZCONOE job.

The OLA Enabler will not copy any panels that you have modified or added to your Classic system. Therefore you will need to add any modified or added panels as SELECT statements so they are copied to your OLA system.

- f Tailor the remainder of the JCL to your Installation's requirements.
- 2 Edit the sample configuration ISZCONOE and change COPY ISZCON01 to COPY ISZCONxx, where xx is the unique identifier for your current configuration. Your ISZCONxx member must include the COPY ISZOLA and COPY ISZCLSO statements as specified in the supplied ISZCON01 member.
- 3 Shut down your current Session Manager system.
- 4 Submit the sample job ISZOEJOB (supplied in library .SISZCONF) and check that it completes successfully. See also 'Definitions created by the OLA Enabler' on page 452.

Note If necessary, you will also need to add COPY xxxxxxxx statements to your ISZCONxx member (in library .SISZCCNF) for each modified or added panel.

- 5 After running the OLA Enabler, and adding additional definitions as required, review the space occupied by each PDS(E). Allowing room for expansion, adjust the space allocated to each PDS(E) as appropriate.
- 6 If the OLA Enabler is run for a second Online Session Manager instance then specify these options: `replace`, `onlink` and `onlname`. For details, see 'OLA Enabler parameters' on page 453.

Definitions created by the OLA Enabler

On successful completion, the OLA Enabler will have:

- Created CONFIG member ISZCONxx and SYSTEM members ISZSYSxx and ISZSYSCM for the Online Session Manager instance, and CONFIG member ISZCONBT and SYSTEM member ISZSYSBT for the Batch Session Manager instance.
- Created this PROFILE definition:


```
PROFILE ISZPROLA
AUTOSELECT 8986
SESSION 8986
QUITACTIVE YES
APPLID ISZAOLA
```
- Optionally, if the OLA Enabler is run with option `addola`, created an OLA session and an OLA application (see page 455).
- If `onlinks` is not specified, created two LINK definitions - one for the Online Session Manager instance and one for the Batch Session Manager instance.
- Removed any RETAIN parameters found on any TERMINAL statements or PROFILE statements.
- Optionally, assigned a session type to each application session (see 'Assignment of session types to sessions' below).

Notes

- 1 All configuration logic (for example, %% If . . .) will have been ignored.
- 2 If NOLINKS is not specified, for each additional Session Manager instance, either:
 - re-run the OLA Enabler for the additional instance (specifying the replace option and appropriate batlnk, onlnk, comlnk and onlname values), in which case an additional LINK definition will be generated
 - or
 - use OLA to add a new LINK definition of the type described in 'LINK definitions' on page 145.

Assignment of session types to sessions

For application sessions defined in APPL configuration control statements, the OLA Enabler optionally assigns *session types* (see 'Session types for sessions' in the *Online and Batch Administration* manual), which are used by Session Manager to eliminate duplicate sessions:

- If the OLA Enabler is run with option addsestype (see page 455) then, for each APPL definition, a unique SESTYPE parameter will be added.

How to load a Language Pack

- Session Manager enables you to specify the language in which screen text for product panels, messages, and so on, is displayed at a user's terminal. Language Packs (LPs), which are supplied with Session Manager in library .SISZCONF (and OLA messages in .SISZMSG) contain all the Panels, Scripts, Text and Messages that have been translated. The default LP is defined in the COPY ISZLEN2 statement in the supplied ISZCON02 member, where EN is the language id for English.
- To load a new LP, add an additional COPY ISZLyy2 statement, where yy is the 2-character language id for the new language, to your ISZCON02 member.

OLA Enabler parameters

The OLA Enabler command can be found in member ISZOLAEN, which is supplied in library .SISZCONF. The ISZOLAEN command can be used to load:

- a *complete* set of customer configuration definitions (see page 454),
- *specified* customer configuration definitions (see page 457),
- a Language Pack (see page 459).

Normally, Session Manager would be run in Batch mode and the ISZOLAEN command would either be specified as the INITIAL_CMD or be called by a script running as the INITIAL_CMD.

Load a complete set of customer configuration definitions

This form of the ISZOLAEN command is used to load a *complete* set of customer configuration definitions from a Classic configuration.

A sample job to run the ISZOLAEN command in this mode can be found in member ISZOEJOB in library .SISZCONF (see 'Running the OLA Enabler' on page 451). Before submitting this sample job, review the sample parameters and tailor them to your Installation's requirements.

Format

```
iszolaen xx
        [test]
        [continue]
        [replace]
        [addola(nnn)]
        [addsestype]
        [batlnk(xxx,yyy)]
        [onlnk(xxx,yyy)]
        [comlnk(zzz)]
        [batname(bbb)]
        [onlname(ooo)]
        [nolinks]
        [clear] [clearonly]
        [trace] [traceall]
```

Required parameter

xx

Is the unique identifier for your current configuration.

A value of BT is not allowed as this is used by Batch Administration.

Optional parameters

test

By default, output members will be created. Specify `test` to indicate that the command should run in test (report only) mode, in which case DDNAME CONFIG is used but output members will not be created.

continue

By default, if an error is found then processing will stop. Specify `continue` to indicate that the command should continue processing if an error is found.

replace

By default, apart from a few exceptions (see below) if duplicate members exist then an error condition will be generated.

Exceptions: CONFIG member ISZCONBT, the Batch LINK member, and the two SYSTEM members ISZSYSBT and ISZSYSCM. These members are not replaced but they do not generate an error condition.

Specify `replace` to indicate that the command should replace duplicate members.

Note If the command is run for a second Online Session Manager instance then specify the `replace` option.

addola(*nnn*)

By default, an OLA session and an OLA application will not be created. Specify addola(*nnn*) to indicate that the command should create:

- An OLA session on all profiles, where *nnn* is the session number and *admin* is the 'selection command' associated with the session:

```
SESSION nnn admin APPLID iszaola
```

- An OLA application, where *sss* is either a unique SESTYPE value generated as a result of specifying the addsestype parameter or, if addsestype has not been specified, is the session number *nnn*:

```
APPL iszaola
  INITSCRIPT olasini
  STARTSCRIPT olastart
  INTERNALSESS yes
  DESC "Online Administration"
```

addsestype

By default, a unique SESTYPE value will not be added to APPL statements. Specify addsestype to indicate that the command should add a unique SESTYPE value to all APPL statements. This will restrict all users to just one session for each APPL statement, but should be used if user-customizable 'priority' sessions are to be used (see the section on 'Changing the order of session definitions' in the *Online and Batch Administration* manual) and any users can have a common enduser setting of SESSPRIAPPL = NO (the default).

batlnk(*xxx,yyy*)

Specifies the prefix and suffix for the Batch link ACB name. The default values for *xxx,yyy* are S01B,B. See 'Example' on page 456.

onlnk(*xxx,yyy*)

Specifies the prefix and suffix for the Online link ACB name. The default values for *xxx,yyy* are S011,1. See 'Example' on page 456.

Note If the command is run for a second Online Session Manager instance then specify the onlnk option to identify the prefix and suffix for the new Session Manager link ACB name.

comlnk(*zzz*)

Specifies the 'common link' literal which is placed between the prefix and suffix for each link ACB name. The default value for *zzz* is T0. See 'Example' on page 456.

batname(*bbb*)

Specifies both the Batch LOCALNODE name and the Batch LINK name. The default value for *bbb* is BATCH. See 'Example' on page 456.

onlname(*ooo*)

Specifies both the Online LOCALNODE name and the Online LINK name. The default value for *ooo* is ONLINE1. See 'Example' on page 456.

Note If the command is run for a second Online Session Manager instance then specify the onlname option to identify a new Session Manager local node name.

`nolinks`

By default, two LINK statements will be created by the Enabler process. If `nolinks` is specified then no LINK statements will be produced regardless of what is specified on the `batlnk`, `onlnk`, `comlnk`, `batname` and `onlname` parameters above. This option should be used if you plan to run the session manager in a Sysplex-only environment. Also review 'Networking and Sysplex considerations (if required)' on page 127 and 'Parallel Sysplex support' on page 387.

`clear`

By default, customer PDS(E)s will not be cleared before loading. Specify `clear` to indicate that the command should clear all customer PDS(E)s before loading.

`clearonly`

By default, operations are performed which allow the OLA-enabled configuration to be implemented. Specify `clearonly` to indicate that the command should not perform these operations, but should clear all PDS(E)s.

`trace`

(This parameter should only be used under the direction of your local Support Representative.) By default, trace (debugging) output is not produced. Specify `trace` to indicate that the command should run in 'light' trace mode.

`traceall`

(This parameter should only be used under the direction of your local Support Representative.) By default, debugging output is not produced. Specify `traceall` to indicate that the command should run in 'heavy' trace mode.

Example

(LINK statements will only be created if `nolinks` is not specified.) If the optional parameters below have the values indicated (their default values)

```
batlnk(S09,BAT)
onlnk(SX9,ON1)
comlnk(SD)
batname(BATCH)
onlname(ONLINE1)
```

then you will need to define an ACB named S09SDON1 in the VTAM subarea where the Batch job is to be executed, and an ACB named SX9SDBAT in the Session Manager VTAM subarea.

Also refer to the `nolinks` parameter on page 456.

On successful completion, the OLA Enabler will have created two LINK definitions – one for the Batch Session Manager instance (BATCH) and one for the Online Session Manager instance (ONLINE1), and several other configuration definitions. For details, see 'Definitions created by the OLA Enabler' on page 452.

Note Since each instance of Session Manager (BATCH and ONLINE1 in this case) loads *all* LINK definitions, the LINK definitions' ACBs contain substitution variables which are assigned appropriate values in the ISZCONxx configuration members. For more information, see 'ISZCONxx definitions' on page 143.

The table below shows the recommended set-up for one instance and Batch:

Proc/job	ISMSTC01	ISMBATCH
Config	ISZCON02	ISZCONBT
Local node	Online1	Batch
Main acb	ISZSMGR	none
gc_acb_pre	SX9	S09
gc_acb_suf	ON1	BAT
gc_common	SD	SD
Opens acbs	SX9SDBAT	S09SDON1
Links	BATCH <-----> ONLINE1 (via SX9SDBAT) (via S09SDON1)	

The table below shows the recommended set-up for two instances and Batch:

Proc/job	ISMSTC01	ISMBATCH	ISMSTC02
Config	ISZCON02	ISZCONBT	ISZCON03
Local node	Online1	Batch	Online2
Main acb	ISZSMGR	none	ISZSMGR2
gc_acb_pre	SX9	S09	S29
gc_acb_suf	ON1	BAT	ON2
gc_common	SD	SD	SD
Opens acbs	SX9SDBAT	S09SDON1	S29SDBAT
	SX9SDON2	S09SDON2	S29SDON1
Links	BATCH <-----> ONLINE1 (via SX9SDBAT) (via S09SDON1)		
		ONLINE2 <-----> BATCH (via S09SDON2) (via S29SDBAT)	
	ONLINE2 <-----> ONLINE1 (via SX9SDON2) (via S29SDON1)		

Load specified customer configuration definitions

This form of the ISZOLAEN command is used to load *specified* customer configuration definitions, samples or scripts to the datasets associated with an *existing* OLA-enabled configuration. The new definitions may be loaded from specified members of either:

- a Classic configuration
- or
- an OLA-enabled configuration.

A sample job to run the ISZOLAEN command in 'load' mode can be found in member ISZLOJOB in library .SISZCONF (see 'Example' on page 459).

Notes

- 1 The configuration definitions to be loaded may be in Classic abbreviated, free-form format, or in OLA-enabled full length parameter format.
- 2 These statements are ignored so are *not* loaded: AUDITROUTE, INSTALLSU, OPTION, SYSTEM and TRACEROUTE.
- 3 Definitions that the OLA Enabler regards as 'system' will *not* be loaded. To load these, rename them, and then after they have been loaded rename them back.
- 4 Any definitions loaded in this way whilst Session Manager is running will *not* be available automatically in the running system. To pick up the updates, you need to either run a Batch update (see 'Batch Administration' on page 55) or issue the PUPDATE command (see the *Technical Reference*).

Format

```
iszolaen load(ddname [membername|mask])
           [test]
           [continue]
           [replace]
           [trace] [traceall]
```

Required parameter

ddname

Is the DDNAME associated with the dataset to be used.

Optional parameters

membername

Indicates that only the definitions in the specified member should be loaded.

Note If you do not specify *membername* or *mask* (see below) then *all* members of the allocated dataset are loaded.

mask

Specifies a mask value to limit selection against an 8-character member name – only definitions in members that match the mask will be loaded.

Note If you do not specify *membername* (see above) or *mask* then *all* members of the allocated dataset are loaded.

These special characters can be included in the mask:

- * An asterisk indicates any number of any characters.
- + A plus sign indicates any single character in that position.

test

By default, output members will be created. Specify *test* to indicate that the command should run in test (report only) mode, in which case the specified DDNAME is used but output members will not be created.

`continue`

By default, if an error is found then processing will stop. Specify `continue` to indicate that the command should continue processing if an error is found.

`replace`

By default, if duplicate members exist then an error condition will be generated. Specify `replace` to indicate that the command should replace duplicate members.

`trace`

(This parameter should only be used under the direction of your local Support Representative.) By default, `trace` (debugging) output is not produced. Specify `trace` to indicate that the command should run in 'light' trace mode.

`traceall`

(This parameter should only be used under the direction of your local Support Representative.) By default, debugging output is not produced. Specify `traceall` to indicate that the command should run in 'heavy' trace mode.

Example

After having OLA-enabled your Session Manager configuration, suppose you decide that you would like to use the sample Mail system. To load the various definitions that represent the Mail system, which are contained in member `ISZC09MA` in library `.SISZCONF`, complete these steps:

- 1 A sample job to run the OLA Enabler command in 'load' mode can be found in member `ISZLOJOB` (in library `.SISZCONF`). Edit the JCL for this sample job so that the `ISZOLAEN` command is as follows:

```
ISZOLAEN LOAD(CONFIG ISZC09MA)
```

- 2 Ensure that:

- the DDNAME to be used, `CONFIG`, is associated with the shipped configuration dataset that contains the `ISZC09MA` member
 - and
 - the OLA datasets are allocated to the appropriate DDNAMEs
- and then submit the JCL.

Load a Language Pack Language packs are supplied by IBM and installed via the SMP/E process.

Setting up OLA for single and multiples Session Manager instances

Please refer to page 141.

Creating the Session Manager started task

After you have successfully set up OLA to administer either a single Session Manager instance (see page 141) or multiple Session Manager instances sharing a configuration (page 142), create the Session Manager started task.

To do this, complete the following steps. If any problems are encountered, please contact your local Support Representative.

- 1** A sample procedure for an OLA-enabled Session Manager started task can be found in member ISZCOLA, which is supplied in library .SISZCONF.
- 2** Edit the sample procedure ISZCOLA and make these changes:
 - a** Add the OLA-enabled configuration datasets (see ‘Configuration DDNAMEs’ on page 448 and ‘Customer, samples and system configuration datasets’ on page 155).
 - b** Change CONFIG 02 to CONFIG xx, where xx should match your current ISZCONxx member.
 - c** Tailor the remainder of the JCL to your Installation’s requirements.
- 3** Start the tailored ISZCOLA started task and check that the OLA-enabled Session Manager system starts up successfully and that *the configuration is error free*.

APPENDIX A

Structure connection specifications

When the program connects to the User List CF structure it issues the IXLCONN macro with the following parameters.

Note Since the structure will contain active users it isn't defined as a permanent structure. In other words, the structure will be created when the first IBM Session Manager for z/OS node initializes and deleted when the last node closes. For more information see *z/OS MVS Programming: Sysplex Services Reference SA22-7618* and *z/OS MVS Programming: Sysplex Services Guide SA22-7617*.

IXLCONN		X
ACCESSTIME=NOLIMIT,		X
ADJUNCT=YES,	use adjunct data (no data elements)	X
ALLOWALTER=YES,	allow structure ALTER	X
ALLOWAUTO=YES,	system-managed processes allowed	X
ALLOWREBLD=NO,	no user-managed rebuild	X
ANSAREA=(R8),		X
ANSLEN==A(CONA_LEN),		X
CFLEVEL=11,	build level	X
COMPLETEEXIT=(R5),		X
CONDATA=WUPCOND,		X
CONDISP=DELETE,	connection disposition	X
CONLEVEL=WUPLEVEL,	our version	X
CONNAME=WUPMBR,	member name	X
ELEMENTRATIO=0,	no data elements	X
EVENTEXIT=(R6),		X
KEYTYPE=SECONDARY,	entry and secondary keys required	X
LISTHEADERS=1,	just one list	X
LOCKENTRIES=0,	no locks	X
MF=(E,WUPCLIST,COMPLETE),		X

MINCFLEVEL=11,	min CFLEVEL	X
NOTIFYEXIT=(R7),		X
REFOPTION=KEY,	use keys to locate	X
RETCODE=WUPCRETC,		X
RSNCODE=WUPCRETR,		X
STRDISP=DELETE,	structure disposition	X
STRNAME=WUPSTRN,		X
SUSPEND=NO,		X
TYPE=LIST		

Index

Numerics

3270 datastream exits
See Input/output 3270 datastream exits

A

ACB name 189
 ACB parameter
 of SYSTEM statement 64
 ACB RANGE 169
 uses 169
 ACB selection
 ACB RANGE 101, 169
 ACB substitution 101, 168
 and TCT entries 170
 best fit algorithm 170
 Best Fit allocation 178
 direct user input 101, 174
 Initscript 101, 175
 Menu TPSL 101, 174
 multiple APPL statements 101, 171
 next best fit 182
 parallel session 101
 SHAREAPPL 101
 SHAREAPPL ACB range 172
 standard methods 101
 User exit E31 101, 175
 user-level session 101, 172
 ACB subparameter 168
 ACB substitution 101, 168
 uses 168
 Accessibility 475
 ACF 99, 290
 addola option 141
 ADDRDDN parameter 354
 ADMIN panel 293
 Commands 293
 Administrator commands 107
 AFFINITY parameter 383
 APF authorization 62
 APPL statements
 multiple 101
 when to use 104
 Application Builder 434
 Application Builder scripts 121
 Application builder scripts 121
 and remote sessions 435
 Application definition
 major node 64
 minor nodes 64
 Application major node 63
 Application node table 64
 Application session recovery 391, 398,
 399
 Application settings 103
 Application Status Change Exit 211
 APPLID TCP_1 358, 371
 ARMWRAP JCL wrapper 135
 ASCII and EBCDIC translation 371
 Asterisk (*)
 as a wildcard character 190
 Audit log (Sysplex Group) 389, 392
 AUDITMSG script parameter 147
 AUDITOGDG parameter 133
 AUDITROUTE statement 144, 448
 Authorization levels 107
 Authorized scripts 450
 Automatic logon 67
 Automatic Restart Manager
 ARMWRAP JCL wrapper 135
 Defining policy 135
 Deregister step 135
 Implementing 135
 Invoking 135

- Modifying startup JCL to use 135
 - Register step 135
 - Security definitions 135
- Autoscripts 119, 381
- B**
- Basic commands 107
- Batch Administration 53, 55
- batlnk parameter 451, 456
- batname parameter 456
- Best Fit allocation 178
- C**
- Call indicators 202
- CALL parameter 113
- CALLEXIT Invocation Exit Point 213
- CALLEXIT parameter 213
- CEDA 339
- CICS Front-End
 - Setting up and using 337
- Classic configuration 100
- CLOSEACBINACT parameter 413
- CLOSEDISC sub-parameter 426
- Closedown Exit Point 245
- CLOSELOGOFF sub-parameter 426
- CLSDST PASS macro 102
- comlnk parameter 451, 456
- Command scripts 120
- Commands
 - authorization levels 107
 - DLOG 147
 - DSTORE 147
 - DTERM 147
 - DUMP 147
 - list of 107
 - QTASK 147
 - specifying remote 384
 - TRACE 147
 - TTPSL 147
- Common enduser parameters
 - logic diagrams 105
- Common session parameters
 - logic diagrams 106
- COMPRESS parameter 376
- Compression savings 332
- Configuration
 - changing 154
 - file 15
 - how achieved 15, 103
 - how processed 153
 - order of statements 154
 - structure 154
 - use of COPY statements 153
- Configuration File 15, 153
- Configuration statements 103
- Content panel section 112
- Control statements
 - summary 159
 - syntax 108
- Controller (session recovery) 411, 414, 417
- COPY statement 144, 153, 448
- COPYFILE command 64
- D**
- DATA command
 - VT220 361
- Data stream compression 331
- Datastream exits, 3270
 - See* Input/output 3270 datastream exits
- DDNAMEs, list of 448
- Debugging 147
- Define panel section 112
- Define section 112
- DEFMENU parameter (of SYSTEM statement) 54
- Destination applications 176
- Direct user ACB input 101, 174
- Disabled access 475
- DISCONNECT command 225
- DLOG command 147
- DROP_SESSION parameter 306, 313
- DSNPREF= parameter 451
- DSTORE command 147
- DTERM command 147
- DUMP command 147
- Duplicate sessions 53
- Dynamic menus 54, 163, 221
- Dynamic profiles 293
- Dynamic storage 86
- dynamic storage
 - estimating 86
- DYNMLOG sub-parameter 307, 310
- DYNMAUTSTHID sub-parameter 307, 311
- DYNMDROPSESSION 311
- DYNMDROPSESSION sub-parameter 307, 311
- DYNMDropsession sub-parameter 306, 311
- DYNMHIDE sub-parameter 307, 312
- DYNMLOGMAX sub-parameter 307, 311
- DYNMTYPE sub-parameter 307, 311
- E**
- E06 Exit Point 211
- E08 Exit Point 212
- E09 Exit Point 213

E11 Exit Point 215
 E21 Exit Point 217, 422
 E22 Exit Point 221
 E25 Exit Point
 Datastream tracing from 273
 Description of 252, 272
 Hardcopy facility 257, 273, 275, 276
 Using windows 273
 E29 Exit Point 225
 E31 Exit Point 227
 E33 Exit Point 229
 E35 Exit Point 252, 276
 Hardcopy facility 257, 276, 278
 E39 Exit Point 233
 E79 Exit Point 242
 E99 Exit Point 245
 EAS parameter 64
 EBCDIC and ASCII translation 371
 ec21_ruser variable 323
 Eclipse support. 126
 Endscripts 119, 381
 Environscripts 119, 381
 ESM, *see* External Security Managers
 Establishing contact with Session Manager
 67
 ETE 242
 Collecting statistics for 251
 User exits 251
 Exit Control Panel 279
 command script 279
 DRIVER command 279
 invoking 279
 status fields 279
 Exit Driver 279
 Exit routine 202
 Exit scripts 119, 282
 accessing exit parameters 282
 setting return codes 283
 use of ISZCMD 282
 using variables 247
 valid parameters 282
 Exits 197
 EXITWALEN parameter
 of the SYSTEM statement 253
 External Security Managers 99, 107, 290

F

Facilities
 Session Manager as a Front-End 426
 shared userid 319
 Features
 Miser 325
 Network data minimiser 325
 Field types 113
 FOCUS command

VT220 361
 FORCETKO (Force Takeover) option
 415
 FORMATMSG script parameter 147
 Functional Enhancement PTF 71

G

gc_acb_pre variable 457
 gc_acb_prefix variable 144, 145, 194, 448
 gc_acb_suf variable 457
 gc_acb_suffix variable 144, 145, 194, 448
 gc_common variable 457
 GDG dataset support 133
 GETMAIN 86
 GETVIS 86

H

HARDCOPY command 261
 HARDCOPY Facility
 HARDCOPY command 261
 Hardcopy Facility
 E25 Exit Point 257, 273, 275, 276
 E35 Exit Point 257, 276, 278
 Header panel section 112
 Help panel
 OLA 53
 HIDE command
 VT220 361
 HIDE parameter 306, 313
 HIDECMD 369
 High Performance Routine 415
 hlq.SISZAUTH 62
 HOLD command
 VT220 361
 Hummingbird HostExplorer 475

I

Idle time interval 225
 IDLEDISC 224, 288
 IDLELOCK 224, 288
 IDLELOGOFF 224, 288
 Initialization Exit Point
 aborting startup 208
 INTSCRIPTs 101
 Initscripts 119, 380
 Initscripts and ACB selection 175
 Input fields 113
 see also Panel Definition facility
 Input/output 3270 datastream exits
 3270 datastream areas 259
 Characteristics 252
 Common exit conventions 254
 Common exit point facilities 253
 Configuration 253

- Datastream tracing from E25 exit 273
 - E25 Exit Point 252, 272
 - E35 Exit Point 252, 276
 - Entry Parameter List format 254
 - Environmental 253
 - General 253
 - Hardcopy facility 257, 273, 275, 276, 278
 - Issuing Session Manager commands 260, 266
 - Overview 252
 - Register conventions 254
 - Sample exit 261
 - Sample macros 261
 - Setting up an output message 261
 - Storage Obtain/Free routine 258, 263
 - Variable Access routine 259, 265
 - INPUTEXIT parameter
 - of the SYSTEM statement 253
 - Installation
 - coding for extended attributes 64
 - define Session Manager to VTAM 63
 - establishing major node 64
 - establishing minor nodes 64
 - overview 57, 71
 - PARSESS parameter 64
 - securing current system 77
 - supplied logmode table 65
 - updating ISZAPPL 63
 - VTAM alterations 63
 - Installation from media 62
 - Installing
 - Session Manager 57, 71
 - INSTALLSU statement 144, 448
 - Instance (session recovery) 411
 - Internet address 352
 - symbolic names 353
 - Internet Protocol 351
 - IPv4 internet address 352
 - IPv6 internet address 352
 - Irrecoverable terminal error 225
 - ISZC09MA supplied member 459
 - ISZCCJOB supplied member 59
 - ISZCINIT configuration member 144, 448, 450
 - ISZCMD 369
 - ISZCMDA 370
 - ISZCOJOB supplied member 59
 - ISZCOLA supplied member 460
 - ISZCOMON configuration member 144, 448, 450
 - ISZCON01 configuration member 59
 - ISZCON02 457
 - ISZCON02 configuration member 59
 - ISZCON03 457
 - ISZCONBT 457
 - ISZCONBT configuration member 143, 452, 454
 - ISZCONOE supplied member 451
 - ISZCONxx configuration member 143, 448, 449, 452
 - ISZCONxx members 51
 - ISZCPnnn configuration member 450
 - ISZCSnnn configuration member 450
 - ISZCSTRT 63
 - ISZE21GA 395, 396, 397
 - ISZE21PH exit 290, 293
 - ISZE21SF exit 293
 - ISZE21VM exit 293
 - ISZE22VM exit 293
 - ISZE25CM supplied member 261
 - ISZE31PT user exit 316
 - ISZE33NY exit 251
 - ISZE79NY exit 251
 - ISZECMD macro 266
 - ISZEENT macro 262
 - ISZEGCIA macro 271
 - ISZEIPB macro 268
 - ISZEOMA macro 270
 - ISZEPLST macro 267
 - ISZEPTKT user exit 316
 - ISZERCAB macro 269
 - ISZERECS macro 272
 - ISZERET macro 263
 - ISZESTOR macro 263
 - ISZEVAR macro 265
 - ISZEXT00 280
 - ISZFHAGE CICS program 339
 - ISZLENPU sample menus 54
 - ISZLOJOB supplied member 458, 459
 - ISZOEJOB 446
 - ISZOEJOB member 58
 - ISZOEJOB supplied member 451, 454
 - ISZOLAEN supplied member 451, 453
 - ISZRSVRD configuration member 450
 - ISZSPSTK PassTicket script member 316
 - ISZSVTM 189
 - ISZSYSBT configuration member 452
 - ISZSYSBT configuration member 454
 - ISZSYSCM configuration member 143, 144, 449, 452, 454
 - ISZSYSxx configuration member 143, 144, 449, 452
 - ISZSYSxx member 104
 - ISZVMODE 65
 - making available to VTAM 65
 - ISZVMODE parameter 64
- J**
- Jaws 475

L

Language Pack
 adding, Classic configuration 60, 73, 115
 adding, OLA configuration 60, 75, 115, 447
 description of 115, 453
 loading 459
 restrictions 116
 using 116
 LINK definitions 452
 LINK definitions, and Sysplex 143, 145
 LINK definitions, setting up 145, 194
 LINK Statement 137
 Linkage conventions 247
 Links
 defining 139
 Literal fields 114, 147
see also Panel Definition facility
 LNKAUTH 62
 LNKLST 62
 LOADDSN= parameter 451
 LOCALNODE parameter 137
 LOGAPPL VTAM parameter 427
 Logmode entry name tables 178
 Logmode entry names 178
 Logmode entry selection 184
 3270 extended features 184
 and remote terminals 184
 application session 185
 Logmode table 64, 65
 ISZVMODE 65
 making available to VTAM 65
 Logmodes 359
 LOGOFF command 225
 LOGOFF SIGNON command 429
 Logon
 automatic 67
 Logon data 68
 Logon Exit Point 215
 Logon procedures 67
 LU Definition Control Vector 186
 LU name
 used in shared userids 320

M

Macros

ISZECMD 266
 ISZEENT 262
 ISZEGCIA 271
 ISZEIPB 268
 ISZEOMA 270
 ISZEPLST 267
 ISZERCAB 269
 ISZEREGS 272

ISZERET 263

ISZESTOR 263

ISZEVAR 265

Masking 189

Members, configuration

ISZCINIT 144, 448, 450

ISZCOMON 144, 448, 450

ISZCONBT 143, 452, 454

ISZCONxx 143, 448, 449, 452

ISZCPnnn 450

ISZCSnnn 450

ISZRSVRD 450

ISZSYSBT 452, 454

ISZSYSCM 143, 144, 449, 452, 454

ISZSYSxx 143, 144, 449, 452

Members, supplied

ISZC09MA 459

ISZCCJOB 59

ISZCQJOB 59

ISZCOLA 460

ISZCONOE 451

ISZE25CM 261

ISZLOJOB 458, 459

ISZOEJOB 451, 454

ISZOLAEN 451, 453

Menu TPSL and ACB Selection 174

Menus

dynamic 54, 163

sample 54

troubleshooting 147

Menus, dynamic 221

Message variable substitution 202

Minor nodes 64

Miser 325

and CPU usage 328

and the Shared Userid facility 322

data traffic reduction 325

determining inbound savings 326

inbound data streams 326

incompatibility 327

outbound data streams 326

restrictions 330

selective implementation 328

statistics 328

use with COMPRESS 332

See also Network data minimiser

MISER parameter 376

MNPS 410, 414

Modified data tag (MDT) 326

MTS parameters 186

Multi-Node Persistent Sessions 410

multiple APPL statements 171

Multiple configuration changes 53

Multiple Exit Driver 279, 292

Exit Control Panel 279

- ISZEXT00 280
- module name standard 280
- operation 280
- restricting update authority 280
- setting up 280
- update request commands 279
- Multiple Session Manager instances 51
- Multiple sessions 100
- Multiple users
 - with same userid 319

N

- Netman 138
- NETSPY 242
 - Collecting statistics for 251
 - User exits 251
- Network data minimiser
 - Miser 325
- Network Data Minimiser Feature 325
- Network Manager 138
- Network Virtual Terminal 358
- Networking considerations
 - VTAM implications 139
- Networking Feature 137, 374
 - and Autoscripts 381
 - and data compression 376
 - and Endscreens 381
 - and Environscripts 381
 - and Initscreens 380
 - and MISER parameter 376
 - and RMISER parameter 376
 - and Startscripts 381
 - and Termscreens 381
 - defining links 139
 - defining nodes 137
 - establishing remote sessions 375
 - LINK Statement 137
 - LOCALNODE parameter 137
 - Network Manager 138
 - Nodes 137
 - parameter sequence 380
 - QUERY command 377
 - reducing network traffic 375
 - remote commands 384
 - REMOTE parameter 377
 - routing 138
 - script sequence 379
 - terms and concepts 137, 373
 - user affinity 383
- Nodes
 - defining 137
- nolinks parameter 452, 456
- Non-parallel sessions 100

O

- OLA
 - datasets 58, 89
 - DDNAMEs, list of 448
 - deciding whether to use 100
 - Menus, lists and displays, *see* OLA menus, lists and displays
 - Online Help 53
 - Security class 107
 - Session Manager started task 460
 - Setting up, *see* OLA setup
- OLA configuration
 - DDNAMEs, list of 448
 - How achieved 448
- OLA Enabler
 - Parameters 453
 - Running 451
- OLA menus, lists and displays
 - Summary list of 52
- OLA security
 - OLACCLASS attribute 450
- OLA setup
 - ISZCONxx definitions 143
 - LINK definitions 145, 194
 - Multiple Session Manager instances, configuration definitions 143
 - Multiple Session Manager instances, description of 142
 - Single Session Manager instance, configuration definitions 141
 - Single Session Manager instance, description of 141
- OLACCLASS common enduser parameter 450
- olasini script 143
- olastart STARTSCRIPT 143
- Online Administration, *see* OLA
- Online Help 53
- onllnk parameter 451, 456
- onlname parameter 456
- Operator commands 107
- OPTION statement 144, 448
- Ordering profiles 107
- Output 3270 datastream exit
 - See* Input/output 3270 datastream exits
- Output fields
 - and variables 113
- OUTPUTEXIT parameter
 - of the SYSTEM statement 253
- Overtyping 107

P

- Panel and Script Language 16, 118, 152
- Panel Definition 112
- Panel Definition facility

- Input fields 113
- Literal fields 114
- Panel Definition 112
- Panels
 - basic field types 113
 - Content panel section 112
 - Define panel section 112
 - defining command area 112
 - Header panel section 112
 - Input fields 113
 - Literal fields 114
 - Output fields 113
 - Process panel section 112
 - sub-definitions 113
 - tracing 121
 - Trailer panel section 112
- Parallel sessions 100, 101
- Parallel Sysplex 388
- Parallel Sysplex support 387
- Parameter block 247
- PARSESS parameter 64, 100
- PassTickets 315
- PASSWORD parameter 291
- PCONTENT statement 113
- PCOPY statement 144, 153, 448
- Performance Monitoring Routine
 - data collection intervals 438
 - implementing 439
 - Performance Statistics Panel 438
 - PERFPAN command 438
- Performance Statistics Panel 438
 - refreshing the panel 438
- PERFPAN command 438
- PHEADER statement 113
- Planned maintenance 398
- Planned switches 398
- Plus (+)
 - as a wildcard character 190
- PPROCESS statement 113
- Prerequisites 61
- Primary applications 176
- Process panel section 112
- Process section 112
- PROFILE statement
 - Examples 109
 - SIGNONPANEL parameter 290
 - when to use 104
- Profiles 103
 - ordering 107
 - settings 103
- PSTKAppl common session parameter 317
- PSTKUser common session parameter 317
- PTFs
 - installing 71

- PTRAILER statement 113
- PUPDATE command 450

Q

- QTASK command 147
- QUERY command
 - and remote sessions 377
- QUIT command
 - VT220 361

R

- RACF 99, 290
- RANGE statement 101
- Rapid Transport Protocol 415
- RDO transaction 339
- Read Buffer (RBUF) command 326
 - disabling 327
 - incompatibility 327
- Reconnect users (Sysplex) 389, 395
- Recovery level 398, 412
- Recovery planning 414
- RECOVERYLEVEL parameter 412
- Recursive signon 428
- Remote nodes 352
- REMOTE parameter 138, 377
- Remote Sessions
 - and variables 381
 - establishing 375
- Response Time Monitor 440
 - and TCP/IP 443
 - displaying response times 440
 - responses 440
- Response Time Statistics 443
- RMISER parameter 376
- RTMT1 parameter 440
- RTMT2 parameter 440
- Runlength compression 331
- RUSER statement (and Sysplex) 390

S

- s_dropsess variable 307
- s_hidden variable 54, 307
- s_sequence variable 54
- s_tn3270e variable 372
- s_tn3270e_dev variable 372
- S09SDON1 456, 457
- S09SDON2 457
- S29SDBAT 457
- S29SDON1 457
- Sample menus 54
- Sample User Exits 292
- Screen reader 475
- Script Facility
 - Application Builder 434

- application builder scripts 121
- autoscripts 119
- endscripts 119
- environscripts 119
- exit scripts 119
- initscripts 119
- startscripts 119
- termscripts 119
- windows scripts 120
- Scripts
 - Application Builder 121
 - categories 118
 - Command 120
 - Exit 119, 282
 - overview 118
 - sequence on remote sessions 379
 - session categories 119
 - tracing 121
 - troubleshooting 147
 - Window 120
- Scripts, authorized 450
- Secondary applications 176
 - session termination 176
- Security 217
 - VSAM based 293
- Security exits
 - Examples 109
- Security levels 107
- Selective signon 428
- SEND command 384
 - and command scripts 384
 - valid commands for use with 384
- Session ACB
 - name 380
 - user-level 187
- Session Manager
 - establishing contact with 67
 - installing 57
- Session Manager as a Front-End
 - CLOSEDISC sub-parameter 426
 - CLOSELOGOFF sub-parameter 426
 - forcing a single session 426
 - logmode table implications 430
 - Recursive signon 428
 - Selective signon 428
 - VTAM definition factors 427
- Session recovery 391, 398, 399
- Session Scripts 118, 119
- Session Switch Exit Point 242
- Session types for sessions
 - Assignment automatically 453
- SESSPRI parameter
 - of the USER statement 44
- SESTYPE common session parameter 453
- SETPROG 63
- Setting up OLA, *see* OLA setup
- SETUP command
 - VT220 361
- Setup panel for VT220 364
- SHAREAPPL ACB range 172
 - and CLSDST PASS 172
 - uses 172
- SHAREAPPL parameter 101
- Shared userid Facility 319
 - ec21_ruser variable 323
- SIDLTIME 288
- SIDLTIME timer 232
- Signoff Exit Point 225
- Signon Completion Exit Point 221
- Signon panel
 - bypassing 68
- SIGNON parameter 105, 106
- Signon security system 293
- Signon Validation Exit Point 217, 422
- SIGNONPANEL parameter 290
- SIMRECON
 - common user parameter 429
- SINGLE command
 - VT220 362
- Slave Session Post-Init Exit 229
- Slave Session Pre-Init Exit 227
- Slave Session Termination Exit 233
- SMP/E installation 58
- SPIN command 133
- SPLXLOCUSER function 391
- SPLXLOG function 391
- SPLXNODES function 391
- Standby (session recovery) 411
- Standby controller (session recovery) 411, 414, 415, 417
- STARTTCP command 358
- Started task for Session Manager 460
- Starter program for CICS Front-end 339
- Startscripts 119, 381
- Statistics
 - for MISER 328
- STOPTCP command 358
- Storage
 - Dynamic 86
- Storage requirements 86
 - and panels 87
 - dynamic storage 86
 - estimating 86
 - examples 87
 - GETMAIN 86
 - GETVIS 86
 - Session Manager Program 86
- Storage shortage 87
- Suppressing multiple sessions 426
- SX9SDBAT 456, 457
- SX9SDON2 457

- Symbolic names 353
 - Syntax
 - configuration statements 108
 - Sysplex 388
 - networking 389
 - Sysplex Group
 - audit log 389, 392
 - Sysplex support 387
 - Sysplex, and OLA 143, 451
 - SYSPLEXGroup parameter 388
 - System settings 103
 - SYSTEM statement
 - ACB parameter 64
 - EXITWALEN parameter 253
 - INPUTEXIT parameter 253
 - OUTPUTEXIT parameter 253
 - SIGNON parameter 105, 106
 - SIGNONPANEL parameter 290
 - when to use 104
 - SYSTEMxx member 104
- T**
- t_affinity variable 383
 - t_config_suf variable 144
 - t_result variable 213
 - t_tn3270e variable 357
 - t_tn3270e_name variable 357
 - t_user_acb variable 187, 188, 356
 - t_user_appl variable 187, 188, 356
 - t_user_qual 320
 - TCP/IP
 - ASCII and EBCDIC translation 371
 - domain names 353
 - EBCDIC and ASCII translation 371
 - escape characters 366
 - Internet address 352
 - issuing Telnet commands 366
 - line-by-line mode 358
 - Network Virtual Terminal 358
 - PF keys and Commands 367
 - ports 353
 - Remote nodes 352
 - sending commands 369
 - specifying remote host 353
 - symbolic names 353
 - Telnet application 354
 - translation tables 371
 - using Session Manager with 351
 - TCP/IP feature 351
 - TCP/IP manager
 - starting 358
 - stopping 358
 - TCP/IP ports 353
 - TCP/IP sessions
 - configuring 358, 371
 - Telnet application 354
 - issuing commands 366
 - Telnet Client connections 358
 - TELNET commands 370
 - TELNET panel 360
 - Terminal settings 103
 - TERMINAL statement
 - SIGNON NO parameter 427
 - SIGNONPANEL parameter 290
 - when to use 104
 - TERMLOGMODE parameter 178
 - Termscripts 119, 381
 - The Panel and Script Language *see* TPSL
 - Timeout exits 288
 - Timer expiry 224
 - Timer Interval Exit Point 212
 - TN320E client operation 371
 - TN3270 support 355, 358
 - TN3270E support 355
 - Top Secret 99, 290
 - TPSL 16, 118, 152
 - TRACE command 147
 - TRACEROUTE statement 144, 448
 - Tracing 147
 - Trailer panel section 112
 - Transmission Control Protocol 351
 - Troubleshooting 147
 - TTPSL command 147

U

 - Unauthorized personnel 217
 - UNBIND WAIT applications 176
 - waiting for a BIND 176
 - UNBIND with BIND forthcoming 176
 - UNBINDAPPL parameter 176
 - Uniform Resource Locator 358, 371
 - UPDATE command 153, 154, 202
 - URL 358, 371
 - User affinity 383
 - AFFINITY parameter 383
 - User commands 107
 - User Exit 197
 - address 1 206
 - address 2 206
 - address 3 207
 - Application Status Change Exit 211
 - call indicators 202
 - CALLEXIT Invocation Exit Point 213
 - checking signon parameters 202
 - Closedown Exit Point 245
 - defining messages 202
 - exit conventions 205
 - exit routine 202
 - invocation points 200
 - linkage conventions 205, 247

- loading 202
 - loading via UPDATE command 202
 - Logon Exit Point 215
 - Message variable substitution 202
 - modifying config statements 202
 - modifying session parameters 227
 - parameter list 206
 - register conventions 205
 - Replacement Exit Point 241
 - return codes 205
 - Session Switch Exit Point 242
 - session termination 233
 - Signoff Exit Point 225
 - Signon Completion Exit Point 221
 - Signon Validation Exit Point 217, 422
 - Slave Session Post-Init Exit 229
 - Slave Session Pre-Init Exit 227
 - Slave Session Termination Exit 233
 - specifying on OPTION statement 202
 - Timer Interval Exit Point 212
 - Variable Access 246
 - Variable Access Routine 206
 - User exit E31 101
 - User exits 60
 - User reconnection (Sysplex) 389, 395
 - User settings 103
 - USER statement
 - Examples 108
 - PASSWORD parameter 291
 - when to use 104
 - User-level session ACB 172, 187
- V**
- Variable Access 246
 - Variable Access Routine 206
 - access return code 250
 - routine input 247
 - routine output 249
 - Variable name 248
 - Variable parts of messages 202
 - Variables
 - gc_acb_prefix 144, 145, 194, 448
 - gc_acb_suffix 144, 145, 194, 448
 - s_hidden 54
 - s_sequence 54
 - t_config_suf 144
 - Virtual Terminal Masking 189
 - VSAM based security 293
 - VT220
 - CURSOR command 361
 - DATA command 361
 - FOCUS command 361
 - HIDE command 361
 - HOLD command 361
 - SETUP command 361, 362
 - ZOOM command 362
 - VT220 Setup 364
 - VT220/VT100 Emulation 360
 - VTAM ACB definition
 - PARSESS parameter 100
 - VTAM alterations 63
 - VTAM application session recovery 391, 398, 399
 - VTAM Definition disk 64
 - VTAM implications
 - and networking 139
 - VTM 189
 - VTM (non-Sysplex) 193
 - VTM (Sysplex) 195
 - VTM and Sysplex environment 193
 - VTM maintenance (and Sysplex) 390
- W**
- Window scripts 120
 - WindowEyes 475
 - Windows feature 120
 - Windows scripts 120
 - Workload Management 388
- Z**
- ZOOM command
 - VT220 362

Bibliography

IBM Session Manager library

The following publications contain information about IBM Session Manager.

	<i>Installation and Customization</i>	GC34-7146-00
	<i>Technical Reference</i>	SC34-7147-00
	<i>User and Administrator</i>	SC34-7150-00
	<i>Panels, Scripts and Variables</i>	SC34-7148-00
	<i>Messages and Codes</i>	GC34-7152-00
	<i>Quick Reference</i>	SC34-7151-00
	<i>Online and Batch Administration</i>	SC34-7149-00
	<i>Program Directory</i>	GI13-0564-00

Accessibility

Accessibility for people with disabilities

The following features make it easier for disabled people to use Session Manager:

- Operation by keyboard alone
- Optional font enlargement
- High-contrast display settings
- Can be used with screen readers
- Absence of audio prompts.

Changing font, color and display settings

Session Manager can be controlled using a 3270 emulator such as IBM Personal Communications or Hummingbird HostExplorer. Refer to the emulator documentation for guidance on adjusting font and color settings.

Using Session Manager with a screen reader

Screen readers can be used to provide accessible output for blind users. Session Manager has been tested with the following screen readers:

- Jaws version 4.5, using Hummingbird HostExplorer and the script file for Hummingbird HostExplorer
- WindowEyes 4.2, using Hummingbird HostExplorer and the set file for Hummingbird HostExplorer.

Contact the screen reader manufacturer for information about the availability of set and script files.

Documentation

Softcopy PDF documentation is shipped with Session Manager. The documentation supports optional font enlargement, high-contrast display settings, and may be operated by the keyboard alone. Alternative text is not provided for screen-reader users. Fully accessible softcopy documentation, with alternative text for diagrams, will be made available on request. Contact your IBM service representative for information.

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply in the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licenseses of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM United Kingdom Laboratories, MP151, Hursley Park, Winchester, Hampshire, England, SO21 2JN. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol ([®] or [™]), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at <http://www.ibm.com/legal/copytrade.shtml>

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

Sending your comments to IBM

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book.

Please limit your comments to the information in this book and the way in which the information is presented.

To ask questions, make comments about the functions of IBM products or systems, or to request additional publications, contact your IBM representative or your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

You can send your comments to IBM in any of the following ways:

- By mail, to this address:
User Technologies Department (MP095)
IBM United Kingdom Laboratories
Hursley Park
WINCHESTER,
Hampshire
SO21 2JN
United Kingdom
- By fax:
 - From outside the U.K., after your international access code use 44-1962-816151
 - From within the U.K., use 01962-816151
- Electronically, use the appropriate network ID:
 - IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
 - IBMLink™ : HURSLEY(IDRCF)
 - Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

- The publication title and order number
- The topic to which your comment applies
- Your name and address/telephone number/fax number/network ID.



GC34-7146-00