z/OS
2.5

TSO/E General Information

SA32-0979-50

Note

Before using this information and the product it supports, read the information in “Notices” on page 63.

This edition applies to Version 2 Release 5 of z/OS® (5650-ZOS) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2021-09-30

Contents

Figures

Tables

About this document

This document supports z/OS (5650-ZOS).

This document describes the facilities of the TSO/E element of z/OS.

Who should use this document

This document is intended for any person who wants an overview of TSO/E.

How this document is organized

This document contains the following chapters:

- Chapter 1, “Introduction,” on page 1 explains the purpose and benefits of TSO/E.
- Chapter 2, “End use of TSO/E,” on page 9 describes the functions that are available to end users of TSO/E.
- Chapter 3, “Programming,” on page 17 describes the programming tools that are available to application and system programmers on TSO/E.
- Chapter 4, “Customization,” on page 29 describes how system programmers can customize the functions and services of TSO/E.
- Chapter 5, “Administration,” on page 49 describes how to administer TSO/E and the Information Center Facility, such as adding and maintaining products and user information.
- Chapter 6, “Diagnosis,” on page 55 describes the facilities for diagnosing program errors on TSO/E.

How to use this document

You will probably use this document differently, depending on the tasks you perform on TSO/E. In general, you may want to read the chapters describing the task or tasks that interest you, such as end use, programming, customization, administration, or diagnosis. For overview information, read Chapter 1, “Introduction,” on page 1.

Where to find more information

Please see *z/OS Information Roadmap* for an overview of the documentation associated with z/OS, including the documentation available for z/OS TSO/E.

How to send your comments to IBM

We invite you to submit comments about the z/OS product documentation. Your valuable feedback helps to ensure accurate and high-quality information.

Important: If your comment regards a technical question or problem, see instead “If you have a technical problem” on page xi.

Submit your feedback by using the appropriate method for your type of comment or question:

Feedback on z/OS function
If your comment or question is about z/OS itself, submit a request through the IBM RFE Community (www.ibm.com/developerworks/rfe/).

Feedback on IBM® Documentation function
If your comment or question is about the IBM Documentation functionality, for example search capabilities or how to arrange the browser view, send a detailed email to IBM Documentation Support at ibmdocs@us.ibm.com.

Feedback on the z/OS product documentation and content
If your comment is about the information that is provided in the z/OS product documentation library, send a detailed email to mhvrcfs@us.ibm.com. We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information.

To help us better process your submission, include the following information:

- Your name, company/university/institution name, and email address
- The following deliverable title and order number: z/OS TSO/E General Information, SA32-0979-50
- The section title of the specific information to which your comment relates
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive authority to use or distribute the comments in any way appropriate without incurring any obligation to you.

IBM or any other organizations use the personal information that you supply to contact you only about the issues that you submit.

If you have a technical problem

If you have a technical problem or question, do not use the feedback methods that are provided for sending documentation comments. Instead, take one or more of the following actions:

- Go to the IBM Support Portal (support.ibm.com).
- Contact your IBM service representative.
- Call IBM technical support.

Summary of changes

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

Note: IBM z/OS policy for the integration of service information into the z/OS product documentation library is documented on the z/OS Internet Library under IBM z/OS Product Documentation Update Policy (www-01.ibm.com/servers/resourcelink/svc00100.nsf/pages/ibm-zos-doc-update-policy?OpenDocument).

Summary of changes for z/OS TSO/E General Information for Version 2 Release 5 (V2R5)

The following content is new, changed, or no longer included in V2R5.

New

The following content is new.

- None

Changed

The following content is changed.

- None

Deleted

The following content was deleted.

- None

Summary of changes for z/OS Version 2 Release 4 (V2R4)

The following changes are made for z/OS Version 2 Release 4 (V2R4).

New

None.

Changed

None.

Deleted

- z/OS support for Server-Requester Programming Interface (SRPI) and TSO/E Enhanced Connectivity Facility are discontinued. The information about Server-Requester Programming Interface and Enhanced Connectivity Facility are removed. This includes that the MVSSERV command is no longer issued.

Summary of changes for z/OS Version 2 Release 3 (V2R3)

This information contains no technical changes for this release.

Chapter 1. Introduction

TSO/E is a base element of z/OS.[1] TSO/E allows users to interactively share computer time and resources. In general, TSO/E makes it easier for people with all levels of experience to interact with the MVS system.

Users of TSO/E

TSO/E has advantages for a wide range of computer users. TSO/E users include system programmers, application programmers, information center administrators, information center users, TSO/E administrators, and others who access applications that run under TSO/E. The different chapters of this document describe the major tasks that each of these users can perform using TSO/E.

Highlights of TSO/E

The TSO/E element of z/OS originated from the former TSO/E component of previous MVS and OS/390 systems. It originally contained a set of commands and a set of system services that users and programmers could use when writing their own commands. During its life, it was continually enhanced and became the primary user interface to the OS/390 system and now to the z/OS system.

The highlights of TSO/E follow. Later chapters describe these highlights in more detail under the user tasks to which they apply.

- The Information Center Facility

  The Information Center Facility is the foundation for building an MVS-based information center (IC). An IC increases user productivity and the computer effectiveness by providing easy-to-use computing tools, data access, education, and other assistance for users who have little or no data processing experience. The Information Center Facility eases users into the data processing environment by providing a series of conversational panels. These panels eliminate numerous command-driven interactions between the user and the system.

  In addition to user services, the Information Center Facility provides panels that enable an administrator to maintain the facility, enroll users, and add, modify, and delete products. TSO/E provides several new functions that help an administrator to easily install, maintain, and upgrade products in the Information Center Facility.

  TSO/E also provides support for tailoring Information Center Facility panels, functions, and environments to the needs of different departments or user groups and individual users. This enhancement eliminates the requirement to make all products in the Information Center Facility available to all users.

- The Session Manager

  The TSO/E Session Manager is an interface to line mode TSO/E. It saves the commands that you enter and the responses that you receive and allows you to redisplay or print them. You can correct or change a command that is displayed on the screen without having to retype the entire command. By allowing you to redisplay, change, and reuse your input, the Session Manager makes TSO/E easier to use.

- Commands

  TSO/E provides numerous commands for both users and programmers that allow them to interact with TSO/E and the MVS system. The ALLOCATE, FREE, and EDIT commands are examples of commands that allow users to manage their data sets. The TEST and TESTAUTH commands let programmers test assembler language programs, including Command processors, APPC/MVS transaction programs,

[1] Before z/OS, TSO Extensions (TSO/E) was an element of OS/390® and was a licensed program for the MVS™ and MVS/ESA System Products. It also was an extension of the Time Sharing Option (TSO) of former MVS systems.

and other programs written in assembler language. The CONSOLE command lets users with CONSOLE command authority perform MVS operator activities from a TSO/E session.

Table 1 on page 13 provides brief descriptions of the TSO/E commands for users. Descriptions of other commands are provided in the individual chapters that discuss the related tasks.

- Online help

  Terminal users can obtain online help for most TSO/E commands. Information Center Facility users can obtain help for each panel and message. In addition, the HELP facility is enhanced to allow installations greater flexibility in adding help information.

  Installations can also provide online help information to users in different languages.

- Data and notice handling

  TSO/E simplifies how data and notices are sent and received. For example, the TRANSMIT and RECEIVE commands let users send data and messages to other users in a network.

  TSO/E improves the LISTBC command so that it requires fewer I/O operations to list the contents of the broadcast data set. The broadcast data set or individual user logs contain messages that either the system or another user sends using the SEND command. In addition, a recovery routine prevents broken mail chains that could occur when message handling is interrupted.

  Notices are also handled more efficiently during logon processing. TSO/E keeps a copy of notices in storage, thereby reducing the I/O operations needed to inform users of waiting messages when they log on.

- Logon processing

  TSO/E provides a full-screen logon panel that makes the logon process easier by:

  – Saving user attributes from one session to the next
  – Allowing program function keys to be used during logon
  – Allowing users to enter commands during logon
  – Explaining the error when incorrect information is specified.

  In addition to the full-screen enhancements, a user can request an expanded private area (region) during logon. The LOGON and ACCOUNT Command Processors allow users to request private areas of up to 2,096,128 K bytes for each terminal session. Your installation can also customize the logon panel, the logon help panel, and customize logon processing using different exits.

- Language enablement

  TSO/E takes advantage of the MVS message service 4 to allow installations to provide TSO/E messages and the TRANSMIT full-screen panel to users in different languages. The TSO/E CONSOLE command also supports the display of translated system messages issued during a console session.

  Several enhancements are provided for language enablement. The logon authorized pre-prompt exit and the PROFILE command support the specification of languages to be used in displaying translated information. Enhancements to SYS1.PARMLIB member IKJTSOxx allow installations to specify help data sets for different languages. The TRANSMIT command is also enhanced to allow users to enter Double-Byte Character Set (DBCS) text on the full-screen panel. Support for logon panels and their help text in different languages is also available.

  The TSO/E REXX external function, SYSVAR, provides support for new arguments that REXX execs can use to obtain language information. Execs can use this information together with the new SETLANG function to set the language in which REXX messages are displayed.

- Security

  Installations that have RACF® installed can use security labels (SECLABELs) to protect system resources. TSO/E provides several enhancements to support the use of security labels. The LOGON command and full-screen logon panel support the specification of a security label to be associated with a user's TSO/E session. SUBMIT command enhancements enable users to submit jobs at a security

label that is greater than the one they are currently logged on with. OUTDES command enhancements let users print the job's security label on each page of output.

Installations can also control communication between users to protect the security classification of information. For example, installations can control and audit the use of the SEND command. LISTBC command processing enhancements let installations restrict users from viewing messages for which they do not have the proper security.

The TRANSMIT command supports a new keyword that lets users specify an alternate data set to log transmitted data. This enhancement allows users to log transmitted data at different security labels.

- CLIST language

  The CLIST language is a high-level programming language that lets programmers issue lists of TSO/E commands and JCL statements in combination with logical, arithmetic, and string-handling functions provided by the language. The programs, called CLISTs, can simplify routine user tasks, invoke programs written in other languages, and perform complex programming functions by themselves.

- Restructured Extended Executor (REXX) language support

  TSO/E provides REXX support to MVS users. REXX is a high-level procedures language that enables inexperienced users as well as experienced programmers to write structured programs called REXX execs. You can execute REXX execs in any MVS address space (both TSO/E and non-TSO/E). TSO/E also allows users to write APPC/MVS transaction programs in the REXX language.

  The REXX language allows programmers to perform logical and arithmetic operations and communicate with terminal users. REXX built- in functions increase usability by allowing you to easily perform character manipulation and conversion operations. REXX also provides support for issuing host commands from within a REXX exec.

  TSO/E extends the programming capabilities of REXX by providing TSO/E functions, REXX commands, and programming and customizing services. These facilities allow you to perform additional tasks such as controlling the execution of a REXX exec and customizing how system services are accessed and used. Some of these facilities are available only to REXX execs that execute in the TSO/E address space.

  REXX execs perform functions similar to CLISTs and can call, and be called by, existing CLISTs and other TSO/E programs. Therefore, REXX is an attractive alternative to the CLIST language.

  TSO/E REXX is the implementation of the Systems Application Architecture® (SAA) Procedures Language on the MVS system. By using the instructions and functions defined for the SAA Procedures Language, you can write execs that run in any of the supported SAA environments, such as VM/SP (CMS).

- The TSO/E service facility

  The TSO/E service facility lets TSO/E users execute authorized or unauthorized programs, TSO/E commands, or CLISTs from an unauthorized environment, while maintaining system integrity. TSO/E Release 1 enhances the TSO/E service facility to allow users to optionally bypass some internal processing when accessing an unauthorized command, program, or CLIST from an unauthorized environment. This enhancement can result in a potential performance benefit and it also permits linkage to the ISPEXEC interface of ISPF. In TSO/E Release 2, the command/program invocation platform support, described in *z/OS TSO/E Programming Services* extends the bypassing of internal processing to the initialization and termination environments for the TSO/E service facility. In TSO/E Release 3, the command/program invocation platform support further extends the TSO/E service facility enhancement to include authorized commands and also unauthorized and authorized programs.

- TSO Command Package

  The TSO Command Package provides functions that help to improve productivity. The functions included are:

  – Support for running terminal sessions as batch jobs
  – Automatic saving of data
  – Accounting facility enhancements
  – Defaults for the user-attribute data set

  - Enhancements to several commands
- Support for a REXX compiler

  Installations can take advantage of the REXX compiler support by adding the IBM Compiler and Library for REXX/370 (licensed program number 5695-013) or a functionally equivalent compiler. A REXX compiler communicates with TSO/E using defined interfaces. A compiled REXX exec executes more efficiently because the exec does not need to be interpreted at run time.
- Support for z/OS UNIX

  Installations can use the functions provided by the TSO/E ALLOCATE and FREE commands to manipulate z/OS UNIX files.
- TSO/E Health Checks

  TSO/E provides health checks that are registered automatically during system initialization. These checks run at regular intervals and they can be disabled at any time. They alert installations to potentially serious problems so that corrective action can be taken to limit the impact.

Major benefits of TSO/E

The benefits of TSO/E can be summarized as follows:

- Improved usability

  The Information Center Facility improves TSO/E usability. It assists non-DP users by making it easier to use many services and products. For example, education services provide panels that allow users to register for take, audit, and produce (write) online courses including read abstracts of courses. Users can also register for and read course abstracts for installation courses given in a classroom. With TSO/E, users can also print course abstracts. The Application Manager makes it easier for installations to add applications to the Information Center Facility. TSO/E increases the usability of Application Manager dialogs by allowing administrators to specify different libraries that are required for an application, for example, CLIST and REXX exec libraries. Information Center Facility processing dynamically allocates the required libraries when a user invokes an application. Administrators can also replace applications directly without first deleting the original application and upgrade existing applications that may have been customized.

  TSO/E also improves the usability of the Information Center Facility by allowing departments, user groups, and individual users to create or tailor application definitions for their own use.

  TSO/E provides usability improvements for a variety of other users. TSO/E administrators can use the simplified RACF commands to define interactive users to both TSO/E and RACF. Therefore, they can maintain all of the information about the users in the RACF database. They no longer need to maintain both the SYS1.UADS and the RACF database.

  TSO/E continues to improve usability for users. The TEST and TESTAUTH commands enable programmers to easily test unauthorized programs and APF-authorized programs. The support for language enablement allows users to receive TSO/E information in their national language.

  Applications running in non-TSO/E address spaces can use many TSO/E services and unauthorized commands. This includes writing APPC/MVS transaction programs that access TSO/E services. Applications can use the TSO/E environment service to build and initialize a TSO/E environment.

  TSO/E provides usability enhancements to logon processing. Installations can customize the logon panel and its help text. Installations can process fields on the logon panel using the pre-display and post-display logon installation exits. For installations that support more than one language, logon panels and their help text are also supported in different languages. In addition, LOGON command full-screen processing is enhanced to perform new password verification. Also provided is a post-prompt exit to customize the logon processing after logon prompting has completed.
- Improved productivity

  TSO/E can help improve end-user and programmer productivity. The functions available through the Information Center Facility, the Enhanced Connectivity Facility, the TSO Command Package and

the Session Manager are intended for that purpose. The CLIST language, TSO/E REXX support and TSO/E command enhancements are also intended to allow end-users and programmers to work more productively.

- Improved performance

  TSO/E provides several enhancements that are intended to improve performance. For example, changes to the way TSO/E processes CLISTs improve CLIST performance. Work areas for TSO/E service routines are obtained during logon processing rather than each time they are required, thereby improving performance. Message-handling performance is also improved by decreasing the number of I/O operations needed to use the LISTBC command and to process notices during logon. The use of separate user logs provides a potential performance benefit. Installations can store messages in separate user logs rather than in the broadcast data set. The use of separate logs reduces possible contention that processors could experience when a large number of messages are being stored in or retrieved from the broadcast data set.

  TSO/E continues to provide ways to improve performance. For example, the ALTLIB command lets users dynamically allocate CLIST and REXX exec libraries. Dynamic definition of CLIST and REXX exec libraries reduces the time that it takes to search for these libraries. Use of the MVS SP virtual lookaside facility (VLF) provides a potential performance benefit to CLIST and REXX exec users by helping to reduce data set I/O and DASD contention. Enhancements to TSO/E REXX support and the TSO/E service facility provide a potential performance benefit to users by helping to reduce the system overhead associated with task creation and termination on the invocation of unauthorized and authorized TSO/E commands and programs.

  REXX execs that are allocated to the SYSPROC system level or application level file are compressed when they are stored. The compression of REXX execs provides a potential performance benefit. This enhancement reduces the amount of virtual storage required for processing REXX execs.

  TSO/E processing of the broadcast data set has been enhanced to use the cross-system coupling facility. This enhancement provides a potential performance benefit during logon processing by helping to reduce I/O to the broadcast data set for the display of system notices.

  Enhancements within the ALLOCATE command potentially improve performance by reducing I/O to the VTOC for existing cataloged data sets and by avoiding MVS LOAD processing on most invocations. In addition, the ALLOCATE command reduces the use of MVS services by streamlining its internal processing.

  TSO/E code optimization enhancements within the TSO/E I/O services, REXX processing, CLIST processing, and TSO/E exit processing, reduce the TSO/E working set size.

- APPC/MVS support

  TSO/E also supports writing APPC/MVS transaction programs in the REXX procedural language. The CPICOMM host command environment allows transaction programs written in REXX to be ported across SAA environments. The LU62 host command environment allows you to use specific features of MVS in conversations with transaction programs on other systems.

- Virtual storage constraint relief

  In the MVS environment, TSO/E provides virtual storage constraint relief. Many TSO/E modules and storage areas reside above 16 MB in virtual storage, thereby freeing more virtual storage below 16 MB for users' applications. In addition, whenever possible, new modules and storage areas are located above 16 MB in virtual storage.

- Installation simplification

  TSO/E provides ways to simplify the installation process.

  System programmers can use SYS1.PARMLIB rather than tables (IKJEFTE2, IKJEFTE8, IKJEFTNS, and IKJEFTAP) to specify authorized commands, authorized programs, and the commands that users cannot run in the background. They can use SYS1.PARMLIB to also specify system defaults for several commands. Using SYS1.PARMLIB makes it easier to customize a system because installations need to change only one file. Programmers no longer have to save tables, and assemble and link-edit them each time that they install TSO/E.

TSO/E enhances SYS1.PARMLIB support. System programmers can use SYS1.PARMLIB to set system defaults for additional commands. They can also use the new PARMLIB command to list the current TSO/E system defaults and dynamically update them (with the contents of IKJTSOxx) without having to re-IPL the system.

SYS1.SAMPLIB contains samples that an installation can customize to simplify the installation process. For example, the IKJTSO00 member of SYS1.SAMPLIB contains samples of the statements that you can use in the IKJTSOxx member of SYS1.PARMLIB.

In TSO/E, a table look-up service also helps simplify installation. System programmers can use the table look-up service to determine which commands and programs are authorized and which commands are restricted from use in the background.

- Systems management

  TSO/E improves system management. The new CONSOLE command allows TSO/E users with CONSOLE command authority to issue MVS system and subsystem commands from a TSO/E session and receive responses to those commands. A message retrieval service, GETMSG, and enhancements to TSO/E REXX support allow application programs and REXX execs to also perform MVS operator tasks in the TSO/E environment.

- Systems Application Architecture support

  TSO/E supports Systems Application Architecture (SAA) by providing the procedures language interface elements of REXX in an MVS environment. The CALL command processor is also enhanced to support SAA.

- Improved debugging aids

  TSO/E provides improved debugging aids. For example, the TEST command contains support for debugging programs that use the IBM 3090 vector facility hardware. In TSO/E, the TEST command is further enhanced to allow programmers to debug programs that use access registers. TSO/E also provides a new command, TESTAUTH, that allows programmers to test and debug APF-authorized programs. The TEST and TESTAUTH commands also allow users to test APPC/MVS unauthorized and authorized transaction programs. The Interactive Problem Control System (IPCS) TSODATA verb exit lists additional information related to CLIST variables. In TSO/E, the IPCS TSODATA verb exit is updated to help programmers diagnose problems related to REXX code or the CONSOLE command. New keywords allow users to format REXX environment blocks and the CONSOLE command control block. Tracing functions similar to those available for CLISTs are available for REXX execs.

  TSO/E provides support for tracing installation exits. Programmers can use the OPERATOR SLIP command to trace exit points on the call and return from the exit and display the exit parameter list.

- Additional exits

  TSO/E provides more exits that allow you to modify or extend the processing of many TSO/E commands and functions to suit the needs of your installation. Many of the exits provide a standard exit parameter list to the exit routines receiving control.

Sysplex support

TSO generic resource support

Generic naming of TSO across multiple MVS systems results in:

- Increased availability
- Better workload distribution

For example, when users log on using a generic TSO system name, they will be allocated to an available system within a sysplex by the MVS workload manager to balance the sysplex loading.

To fulfill the TSO Generic Function of TSO/E you must have the following settings in your SYS1.PARMLIB(GRSRNL).

```
/*****************************************************************/
/*  SYSTEM INCLUSION RESOURCE NAME LIST - RNLDEF STATEMENTS      */
/*****************************************************************/

RNLDEF RNL(INCL) TYPE(GENERIC)
QNAME(SYSIKJBC)

RNLDEF RNL(INCL) TYPE(GENERIC)
QNAME(SYSIKJUA)
```

Figure 1. Settings in SYS.PARMLIB(GRSRNL)

For more information about the TSO generic resources, see *z/OS MVS Initialization and Tuning Reference* and also *z/OS MVS System Commands*.

Reconnection support ensures that a user's address space is kept for a specified reconnect interval after a TSO user loses connectivity to the TSO system. During this interval, if the user tries to reconnect, he will be automatically reconnected to the system, application, and address space that he originally had.

SEND command sysplex support

The sysplex enhancements to the SEND command allow:

- TSO/E users to send messages to other users without having to know what system the recipient is logged on to.
- TSO/E also provides the health check, TSOE_SEND_CHECK, to check the current setting for LOGNAME under SEND processing. It reports on whether user logs are being used.

For more information about the SEND command, see *z/OS TSO/E Command Reference*.

- The TSO/E operator to broadcast a message with a single command to all TSO/E users that are:
  - Logged on to any system within a sysplex, or are
  - Logged on to a particular system within a sysplex

For more information about the OPERATOR SEND command, see *z/OS TSO/E System Programming Command Reference*.

PARMLIB command sysplex support

The sysplex enhancements to the PARMLIB command allow the system programmer to:

- UPDATE the TSO/E settings on all or on any subset of systems in the sysplex with a single command.
- LIST the TSO/E settings for all systems in the sysplex, or for any subset of systems in a sysplex, and to show the listing in a compressed fashion.
- Manage PARMLIB members more flexibly because the member IKJTSOxx can reside in any data set that is defined in the PARMLIB concatenation list.
- TSO/E also provides a health check, TSOE_PARMLIB_CHECK, to check whether the default TSO/E PARMLIB settings are being used, in case an error occurs at the initialization time.

For more information about the PARMLIB command, see *z/OS TSO/E System Programming Command Reference*.

Chapter 2. End use of TSO/E

End use means directly using a product for its intended purpose; in this case, using TSO/E to interact with MVS and perform work on the system. This work includes running programs, creating reports, and entering and managing data. The following topics describe the end use facilities that TSO/E provides.

TSO/E Information Center Facility

The Information Center Facility contains two conversational, panel-driven interfaces.

- The end-user interface enables users who have little or no knowledge of data processing to easily access and use products and services. With TSO/E, this interface also enables departments or user groups and experienced end-users to invoke group and private Application Manager dialogs to tailor Information Center Facility menus for their own use.
- The administration interface enables system administrators to maintain the Information Center Facility. For information about administration tasks, see "Administering the TSO/E Information Center Facility" on page 50.

By accessing program products through the Information Center Facility, end users can generate reports, modify and manipulate data, perform business-related analyses, make spontaneous inquiries of the system, and perform other tasks that require the use of computing systems. Installations can customize the Information Center Facility by adding additional products or deleting references to products they do not have. In addition, the Information Center Facility also provides on-line tutorials and help panels for the end user and Information Center Facility administrator.

The Information Center Facility uses the Interactive System Productivity Facility (ISPF) dialog manager services.

The Information Center Facility end user interface begins with a main menu panel that displays a list of options with descriptions and characters for selecting the options. If your installation has not customized the Information Center Facility, your main menu panel for end user tasks will look something like the panel shown in Figure 2 on page 9.

```
                    TSO/E Information Center Facility User Services
 OPTION===>_

 Select one of the following options.  To scroll, press UP or DOWN.

 0  DESCRIBE      - Read a short description of the options on this panel
 1  NEWS          - Obtain system news
 2  NAMES         - Find a name/phone number
 3  OFFICE        - Use mail/document/other office services
 4  PROGRAM       - Use program creation/execution services
 5  ANALYSIS      - Perform decision support/data analysis/
                    report creation
 6  CHART         - Create charts/graphs
 7  COURSES       - Use education services
 8  PDF           - Use ISPF/PDF services
 9  PROBLEM       - Report problems
 10 UTIL          - Information Center Facility Utilities
 I  INTRO         - Learn to use the Information Center Facility
 T  TUTORIAL      - Read a detailed description of the options on this panel
 X  EXIT          - Exit


To view PF key definitions, type KEYS on Command or Option line of any panel.
```

Figure 2. Information Center Facility User's Primary Panel

The Information Center Facility provides the following end use functions:

- News service

- Names directory
- Conduit dialogs (options 3, 4, 5, 6, 8, 9, 10)
- Education services
- Group and private Application Manager
- Group specification
- Print service
- Tutorials and help information

News Service

The news service allows Information Center Facility users to view news items distributed on-line by an administrator. With TSO/E, users can also print news items. Users are notified of added news when they enter the Information Center Facility. They can use the NEW command within the news service to see just that news, or they can request to see news items dated on or after a specified date. Users can also make requests to the administrator to add news items.

Names Directory

The names directory gives Information Center Facility users access to information about other users, such as a person's phone number, user ID, address, and title. The directory can also contain groups. A group can contain names of individuals and names of other groups. Using a group name saves time because a user specifies the group name to represent frequently used combinations of names. Names and groups can be added, updated, and deleted by an Information Center Facility Administrator. The names directory also allows users to maintain and use a private directory as well as the master directory that the administrator maintains.

A user can view private directory entries, master directory entries, or a merged list that contains the names in both the master directory and the private directory. A user can also use the Information Center Facility to ask the administrator to make updates to the master directory.

Conduit Dialogs

On the user's main menu panel, options 3, 4, 5, 6, 8, 9, and 10 support conduit dialogs. Each conduit dialog provides an interactive process for the user to access the associated licensed program product installed at an installation. Using the program products, users can generate reports, modify and manipulate data, perform business-related analyses, make spontaneous inquiries of the system, and perform other tasks that require the use of computing systems.

Education Services

Education services let users take a course, audit a course, produce (write) a course, view a course abstract, and request registration in a course. With TSO/E, users can also print course abstracts.

Users can access IIPS, COMPUTER, and CLASSROOM type courses. The IIPS and COMPUTER courses are on-line courses. An IIPS course is a course that is developed using the Interactive Instructional Presentation System (IIPS) and the Interactive Instructional Authoring System (IIAS). A COMPUTER course is any course, other than an IIPS course, that the user can access on-line. CLASSROOM refers to a course that is conducted in a classroom.

The Information Center Facility administrator can provide abstracts for the courses, register students, keep a record of course registration, and modify the administration defaults to indicate which courses are actually available at an installation.

Group and Private Application Manager

The Information Center Facility is made up of different services that users can access. Installations can define these services to the Information Center Facility using Application Manager dialogs. To be defined by Application Manager, a service must be broken down into parts called applications. After a service is

added, an administrator can copy its applications, modify or delete them, and determine where they are used.

The Information Center Facility supports three levels of Application Manager (system, group, and private). The system level Application Manager dialog is accessed from the Information Center Facility administration panel (option 5, SYSDEF) and is used by system administrators to maintain applications defined for an entire system. The group and private Application Manager dialogs are accessed from the user's primary panel (option 4, PROGRAM) and can be used to create or tailor application definitions to the needs of specific groups of users or individual users.

If you have been assigned as the administrator for your group, you can use group Application Manager to customize the options available on Information Center Facility menus for your group. For example, your department may require a service that is not defined at the system level. You can use group Application Manager to add this service to the Information Center Facility and make it available just for your group. You can also make certain system level services unavailable to your group.

If you are an experienced user and the system or group application definitions do not satisfy your needs, you can use private Application Manager to customize application definitions for your own use. For example, if the system administrator has installed the APL2® program product as an option on the programmer services panel, and you primarily use the Information Center Facility to write APL2 programs, you can use private Application Manager to make APL2 an option on your main menu panel.

See "Applications" on page 51 for an overview of the operations you can perform on applications using Application Manager. See *z/OS TSO/E Administration* for complete information about how to use group and private Application Manager.

Group Specification

Your installation may have tailored Information Center Facility menus for different departments or user groups to satisfy their different needs. If you are given authorization, you can use the menus tailored for a specific group or department. The group specification function, which is accessed from option 4, PROGRAM, allows you to specify the name of a group whose menus you want to use for your next Information Center Facility session. For more information, see *z/OS TSO/E Primer*.

Print Service

The print service, which is available under option 10, UTILITY, allows users to print both sequential and partitioned data sets, with all or selected members. If users are unsure of the data set name or member name, they can display data set and member selection lists from which they can specify a print request. They can also select a printer from the list of available printers and request multiple copies of the printed data set.

Tutorials and Help Information

The on-line tutorial for end users describes how to use the basic functions of the Information Center Facility. End users can access specific topics of interest or view the full tutorial sequentially.

Most of the panels in the Information Center Facility have associated help panels. Help panels provide additional information about a panel and assist the user in making decisions.

Most of the messages in the Information Center Facility have associated help panels. The help panels provide a more detailed explanation and tell the user how to proceed.

Session Manager

TSO/E Session Manager is an interface to line mode TSO/E that provides full-screen display support for line-oriented commands, programs, CLISTs, and REXX execs. TSO/E Session Manager keeps a complete journal of everything that happens during your terminal session while you are in line mode TSO/E. It records everything you type in and everything the system displays. Any time during your terminal session, you can look at work you did in the beginning, middle, or end of your session. TSO/E Session Manager also lets you print a copy of this information. Using Session Manager, you can:

- Keep a record of your interaction with TSO/E.
- Edit and reuse previous input and output with a minimal number of keystrokes. This data can easily be reentered as input, saved in a data set, or printed.
- Access, compare, and manipulate the data needed to do your work directly on the screen. You no longer have to rely on physical documents and listings that are often outdated, time-consuming to obtain, and cumbersome to work with. You can use PF keys to locate data quickly and to move efficiently through data streams.

TSO/E Session Manager enhances the usability of the TSO/E TEST command and other interactive debugging tools. You can keep track of which data areas you have viewed and print the output from the TEST command.

To use Session Manager, you need access to a LOGON procedure that recognizes Session Manager.

The Session Manager default screen supplied by IBM is divided into five major areas called *windows*. These windows can be used to enter, look at, and change the work on the screen. Session Manager Figure 3 on page 12 shows the default display screen.

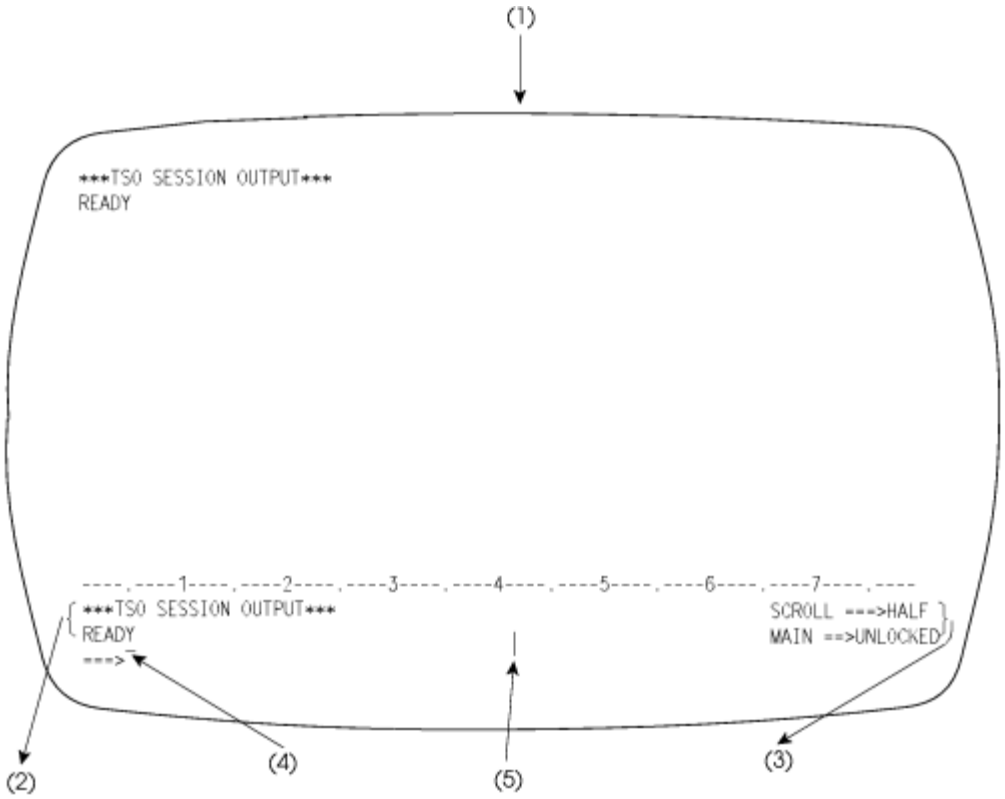

Figure 3. Session Manager Display Screen

The following list describes each window:

Windows
Definition

(1) MAIN
The large portion of the display screen above the numbered line is the MAIN window. This window displays a certain number of lines in the stream of data containing TSO/E input and output. If you wish to submit one or more of these lines as input to TSO/E, type over any character on a line and press the Enter key.

(2) CURRENT
Just below the numbered line is the CURRENT window. When you first log on, this window displays the last two lines of output generated. A PF key lets you alternate the contents of this window between the last two lines of input entered and the last two lines of output generated.

(3) STATUS

The STATUS windows for the MAIN window are located in the lower right hand corner of the display screen. The top window shows the scroll amount (labelled SCROLL===>HALF) and the bottom window indicates whether the MAIN window is locked or unlocked.

The top of the MAIN window contains older data in the stream. When enough data has been generated to fill the MAIN window, the older data begins to be pushed out of the top of the window. You can use PF keys to scroll forward, backward, left, and right in the stream. This action locks the window in place over the desired section of the stream. The keyboard remains unlocked so that you can edit lines in the window.

(4) ENTRY

The ENTRY window begins right after the arrow near the bottom of your display screen and continues to the vertical bar on the last line of the screen. This area is where you normally type in TSO/E commands.

(5) PASSWORD

The window after the vertical bar is the PASSWORD window and can be used to type in passwords for your data sets. The information you type in the PASSWORD window does not appear on the display screen or in your session journal.

The TSO/E Session Manager also uses the PASSWORD window to display any error messages to you.

Installations can tailor the layout of the display screen and the terminal environment to the requirements of its users. Experienced users can also tailor the screen layout and PF key definitions to suit their own needs by using Session Manager commands. These commands can easily be put in a CLIST, enabling all users to tailor the environment for themselves.

End user commands

TSO/E includes a set of commands that you can enter at your terminal to communicate quickly and directly with the system. Table 1 on page 13 lists, in alphabetical order, the TSO/E commands for end users. For more information about what the commands do and how to use them, see *z/OS TSO/E Command Reference*.

Table 1. Summary of the TSO/E commands

Command	Function
ALLOCATE	Dynamically allocates data sets.
ALTLIB	Defines alternative application-level libraries of REXX execs or CLISTs.
ATTRIB	Builds a list of attributes for non-VSAM data sets.
CALL	Loads and executes a program.
CANCEL	Ends the processing of batch jobs submitted at your terminal.
DELETE	Deletes data set entries or members of a partitioned data set.
EDIT	Creates, modifies, stores, submits, retrieves, and deletes data sets. See command definitions for definitions of EDIT subcommands.
END	Ends a CLIST.
EXEC	Executes a CLIST or REXX exec.
EXECUTIL	Changes various characteristics that control how REXX execs execute in the TSO/E address space.
FREE	Releases previously allocated data sets, changes the output of a SYSOUT data set, deletes attribute lists, or changes data set disposition. The FREE command also frees dynamic output descriptors created using the OUTDES command.

Table 1. Summary of the TSO/E commands (continued)

Command	Function
HELP	Gets information about the function, syntax, and operands of commands and subcommands and information about certain messages.
LINK	Invokes the linkage editor service program.
LISTALC	Lists data sets that are currently allocated to the TSO/E session.
LISTBC	Displays messages of general interest.
LISTCAT	Lists entries from a catalog by name or entry type.
LISTDS	Displays attributes of data sets.
LOADGO	Loads a compiled or assembled program into real storage and begins execution.
LOGOFF	Ends your terminal session.
LOGON	Starts your terminal session.
OUTDES	Creates or reuses dynamic output descriptors. (Supported for JES2 only)
OUTPUT	Directs output from a job to your terminal or to a specific data set; deletes the output, changes output class, routes output to a remote workstation, or releases the output for a job for printing by the subsystem.
PRINTDS	Formats and prints data sets on any printer defined to JES.
PROFILE	Changes or lists your user profile.
PROTECT	Prevents unauthorized access to your non-VSAM data sets.
RECEIVE	Retrieves transmitted files and restores them to their original format.
RENAME	Changes the name of a non-VSAM cataloged data set, changes the member name of a partitioned data set, or creates an alias for a partitioned data set member.
RUN	Compiles, loads, and executes the source statements in a data set.
SEND	Sends a message to another terminal user or to the system operator.
SMCOPY	Copies all or part of a stream or data set to another stream or data set.
SMFIND	Locates a string of characters in a stream.
SMPUT	Places a string of characters in a stream.
STATUS	Displays the status of a job.
SUBMIT	Submits one or more batch jobs for processing.
TERMINAL	Lists or changes operating characteristics of your terminal.
TEST	Tests a program or command processor written in Assembler language.
TRANSMIT	Sends information, such as a message or a copy of information in a data set, to another user in the network.
TSOEXEC	Invokes an authorized command from an unauthorized environment.
TSOLIB	Defines alternative load module libraries to a user's session.
VLFNOTE[1]	Notifies the virtual lookaside facility (VLF) that a change has been made to a partitioned data set or a non-partitioned data set.
WHEN	Tests return codes from programs invoked from an immediately preceding CALL or LOADGO command, and to take prescribed action if the return code meets a specified condition.

[1] The VLFNOTE command is an MVS/ESA SP command that TSO/E supports.

Assistance to users

TSO/E provides users with on-line help for commands when they type a ? in response to a prompt. By default, TSO/E provides the prompt mode HELP function for all keyword operands on all commands except the TEST command. It is also available for positional operands of the ATTRIB, CALL, CANCEL, EDIT, EXEC, HELP, OUTPUT, RUN, and SEND commands. The additional information is not available for any subcommands.

TSO/E also provides enhancements to the HELP facility that allow installations to customize help processing. See “HELP data set usage” on page 32 for more information.

Chapter 3. Programming

Programming involves designing, coding, compiling, executing, and testing programs that put the computing system to work for your own purposes. The following topics describe the programming tools that TSO/E provides for the application or system programmer.

REXX programming language

The REstructured eXtended eXecutor (REXX) language is a high-level procedures language that enables inexperienced users as well as experienced programmers to combine REXX instructions and host commands and services into programs called REXX execs. REXX execs can execute in any MVS address space (both TSO/E and non-TSO/E).

The REXX language is particularly suited for:

- Command procedures
- Application front ends
- Macros for ISPF edit
- Prototyping
- Application programs that are to be used in more than one environment
- APPC/MVS transaction programs

For complete information about the topics discussed below, see *z/OS TSO/E REXX User's Guide* and *z/OS TSO/E REXX Reference*.

Features of REXX

The REXX language has several attractive features that make it a powerful programming tool. These include:

- Easy-to-use instructions and flexible syntax and format
- Extensive string manipulation capabilities
- Broad set of built-in functions
- Support for issuing host commands
- Easy testing and execution of REXX execs
- Tracing facilities and the interactive debug facility
- Compatibility with other languages and systems

Instructions, syntax and format

REXX instructions and the flexible syntax and format supported by the REXX language allow programmers to write structured application programs that are powerful and easy to read. Many REXX instructions are common words that aid in the understanding of a program by others. In addition, the REXX language has several *structured programming* instructions, for example, IF, SELECT, and DO WHILE. Because the language does not impose any restrictions on program format, you can combine several instructions on one line or a single instruction can occupy more than one line. An instruction can start in any column, and you can skip spaces in a line or several lines. Programs can, therefore, be coded in a format that emphasizes their structure making them easier to read.

Parsing capabilities

REXX includes extensive parsing capabilities for string manipulation. This feature of REXX allows you to set up a pattern to separate words, groups of characters, or to edit any type of string.

Built-in functions

REXX provides a broad set of built-in functions that perform string manipulation, conversion, and information retrieval operations.

A complementary set of functions support the Double-Byte-Character Set (DBCS), which supports languages (such as Japanese Kanji) that have more than 256 characters, the maximum number that can be represented with one byte of data.

Support for issuing host commands in an exec

Issuing host commands from within a REXX exec is an integral part of the REXX language. For example, in the TSO/E address space, you can use TSO/E commands, ISPF commands, and ISPF EDIT commands in a REXX exec. In both TSO/E and non-TSO/E address spaces, you can invoke MVS programs that use MVS services. You invoke these programs through the host command environments LINK and ATTACH.

With TSO/E, you can also use MVS system and subsystem commands in a REXX exec that executes in the TSO/E address space. You require CONSOLE command authority to use this support.

TSO/E provides new host command environments, LINKMVS, LINKPGM, ATTCHMVS, and ATTCHPGM, which allow you to pass multiple parameters to an invoked module and allow the invoked module to update the parameters.

Testing and executing REXX execs

Because REXX is an interpretive language, you can execute an exec in the same form that you write it; you do not have to compile it first. You can execute execs in any MVS address space (both TSO/E and non-TSO/E) using several different methods.

To test an exec, you can simply execute it, fix any errors, and then re-execute it. REXX also provides several debug facilities that you can use to trace the execution of an exec. See “Debug facilities for REXX execs” on page 56 for more information.

Tracing facilities and the interactive debug facility

REXX provides a TRACE instruction and a TRACE built-in function that you can use to display how the language processor evaluates each operation in a REXX exec. Both the TRACE instruction and TRACE built-in function can also be used to control the interactive debug facility. The interactive debug facility allows you to interactively debug REXX execs in the TSO/E address space from your terminal session. For more information about using the REXX tracing facilities and the interactive debug facility, see “Debug facilities for REXX execs” on page 56.

Compatibility

REXX execs perform functions similar to CLISTs, and can call and be called by existing CLISTs and other TSO/E programs. Therefore, REXX is an attractive alternative to the CLIST language.

TSO/E REXX is the implementation of the Systems Application Architecture (SAA) Procedures Language on TSO/E. By using the instructions and functions defined for the SAA Procedures Language, you can write REXX execs that will run in any of the supported SAA environments, such as VM/SP (CMS). *SAA Common Programming Interface Procedures Language Reference* describes the instructions and functions the SAA Procedures Language offers.

Using APPC/MVS services in a REXX exec

TSO/E REXX supports invocation of APPC/MVS services from the REXX language. APPC/MVS allows transaction programs to be written in the REXX programming language.

- The CPICOMM host command environment allows you to invoke the SAA CPI Communications calls. CPICOMM allows transaction programs written in REXX to be ported across SAA environments. The CPICOMM environment is available in both TSO/E and non-TSO/E address spaces.
- The LU62 host command environment allows you to invoke APPC/MVS calls. LU62 allows you to use specific features of MVS in conversations with transaction programs on other systems. The LU62 environment is available in both TSO/E and non-TSO/E address spaces.
- The APPCMVS host command environment allows APPC transaction programs to invoke the callable services of the APPC/MVS server facility and callable services related to the testing of transaction programs.

Additional TSO/E REXX support

TSO/E extends the programming capabilities of REXX by providing functions, REXX commands, and programming and customizing services that you can use to perform additional tasks. Some of these functions, commands, and services are available only to REXX execs that execute in the TSO/E address space. See *z/OS TSO/E REXX Reference* for complete information about TSO/E REXX support available to REXX execs that execute in TSO/E and non-TSO/E address spaces.

TSO/E functions and REXX commands

In addition to the REXX language, TSO/E provides functions and REXX commands that you can use in an exec. The functions, known as external functions, allow you to perform tasks such as retrieving information about a data set's attributes, and setting the prompt function on or off for TSO/E interactive commands. Several functions are similar to TSO/E CLIST functions and control variables. These functions include LISTDSI, OUTTRAP, PROMPT, SYSDSN, and SYSVAR.

- The GETMSG external function allows you to retrieve system messages issued during a console session. The requested message may be either a solicited message (response to an MVS system or subsystem command) or an unsolicited message that has been routed to your console. You require CONSOLE command authority to use the GETMSG function.
- The SETLANG external function allows you to dynamically set the language in which REXX messages are displayed. This function is available to REXX execs that run in any MVS address space.
- The MVSVAR and SYSCPUS functions return information about various aspects of the sysplex environment, respectively provide information about the number of CPUs and their serial numbers. With these functions, REXX execs can be written that can be shared across systems, nonetheless using system-unique data sets, for example. Both functions are available to REXX execs that run in any MVS address space.

TSO/E also provides support for user-written external functions and subroutines.

The REXX commands allow you to:

- Control or execute I/O processing to and from allocated data sets
- Change characteristics that control how a REXX exec executes
- Check for the existence of a specific host command environment
- Perform data stack requests (see “The data stack” on page 21 for a description of the data stack)

TSO/E REXX programming services

The REXX programming services that TSO/E provides are:

- IRXJCL and IRXEXEC - Exec Processing

The IRXJCL and IRXEXEC routines are programming interfaces to the REXX language processor. You can use these routines to execute an exec in any MVS address space. You can execute an exec in MVS batch by specifying IRXJCL as the program name on the JCL EXEC statement. Any application program, including a REXX exec, in any MVS address space, can call either IRXJCL or IRXEXEC to execute a REXX exec.

- IRXEXCOM - Variable Access

  The IRXEXCOM variable access routine lets unauthorized commands and programs access and manipulate REXX variables.

- External Functions and Function Packages

  You can write your own external functions and subroutines to extend the programming capabilities of the REXX language. You can also group frequently used external functions and subroutines into a *package*, which allows for quick access to the packaged functions and subroutines. If you write external functions or subroutines that you want to include in a function package, you must write them in a programming language that supports the system interfaces for function packages.

- IRXSUBCM - Maintain Host Command Environments

  You can use the IRXSUBCM routine to maintain entries in the host command environment table, which contains the names of the valid host command environments and the routines that handle the execution of commands. A REXX exec can direct a command to any one of these host command environments.

- IRXIC - Trace and Execution Control

  The IRXIC routine is an interface to the REXX immediate commands HI, HT, RT, TE, and TS. A program can call IRXIC to use one of these commands to control the execution and tracing of REXX execs. See “Debug facilities for REXX execs” on page 56 for information about additional trace facilities that TSO/E provides for REXX execs.

- IRXRLT - Get Result

  You can use the IRXRLT routine to obtain the result from an exec that was executed by calling the IRXEXEC routine. You can also use the IRXRLT routine to obtain a larger area of storage to store the result from an external function or subroutine.

- IRXSAY - SAY Instruction

  You can use the IRXSAY routine to write to the output stream. The output stream is typically directed toward the user for display at the workstation, but you can alter the output destination based on the implementation. This routine performs the same functions as the SAY keyword instruction. For additional information about the functions of the SAY instruction, see *z/OS TSO/E REXX Reference*.

- IRXHLT - HALT Condition

  The IRXHLT routine allows application programs to query or reset the HALT condition. This routine provides an interface between programs that require direct access to information on the HALT condition and TSO/E, which maintains the information. For additional information about the HALT condition, see *z/OS TSO/E REXX Reference*.

- IRXTXT - Text Retrieval

  The IRXTXT routine allows application programs to present the date in the same format presented by the REXX interpreter. The day is displayed in the format specified by the Weekday option of the DATE function. The month is presented in the format of either the Month option or the Normal option of the DATE function. See *z/OS TSO/E REXX Reference* for a description of the DATE function and the format of its options.

  The IRXTXT routine also allows you to retrieve syntax error messages provided by TSO/E REXX. You retrieve the error message by specifying the error message number. See *z/OS TSO/E REXX Reference* for error messages and message codes.

- IRXLIN - REXX Line size

  You can use the IRXLIN routine to determine the current terminal line width. This width represents the point at which the language processor breaks lines that are displayed using the SAY instruction or

the IRXSAY routine. This routine performs the same function as the REXX LINESIZE function. See *z/OS TSO/E REXX Reference* for more information about the LINESIZE function.

REXX customizing services

TSO/E provides services you can use to customize REXX processing. Several of these services let you change how an exec is processed and how the language processor interfaces with the system to access and use system services, such as storage and I/O. See "REXX processing" on page 38 for a description of each service that TSO/E provides to customize REXX.

The data stack

TSO/E REXX uses a data stack to store data for use by an exec. The data stack is similar to the VM queue. Using REXX instructions, you can place elements on the stack and remove elements from the stack when they are needed. In addition, there are several TSO/E REXX commands that you can use to perform data stack services. The data stack can be used by REXX execs in both TSO/E and non-TSO/E address spaces.

Storing REXX execs

When you write a REXX exec, you can store the exec in either a sequential or partitioned data set. If you plan to create more than one exec, it is easier to create a REXX library as a partitioned data set (PDS) with execs as members.

If you create a PDS, you can allocate the PDS to a system file. In the TSO/E address space, allocating a PDS containing execs to a system file allows you to:

- Implicitly execute execs by simply entering the member name of the data set that contains the exec.
- Invoke user-written external functions and subroutines written in REXX that are also allocated to a system file.
- Control the search order by concatenating the data sets within the file.

In the TSO/E address space you can allocate a PDS containing execs to either the SYSPROC or SYSEXEC file. In non-TSO/E address spaces, you can allocate a PDS containing execs to the SYSEXEC file only.

Using the SYSPROC and SYSEXEC files

TSO/E provides a system file, SYSEXEC, that can contain REXX execs only. You can use the SYSEXEC file to store execs in both TSO/E and non-TSO/E address spaces.

In TSO/E, the SYSPROC system file can contain REXX execs as well as TSO/E CLISTs. If you store a REXX exec in a PDS that is allocated to SYSPROC, the exec must begin with a comment and the first line of the comment must include the characters "REXX". This is required in order for the TSO/E EXEC command processor, during implicit execution, to distinguish REXX execs from CLISTs, which are also stored in data sets that are allocated to SYSPROC.

If you store your REXX execs in a PDS that is not allocated to SYSPROC, you do not need to begin the execs with a comment. However, it is recommended that you start all execs with a comment in the first column of the first line and include the characters REXX in the comment. In particular, this is recommended if you are writing REXX execs for use in other SAA environments. It is also recommended that you include the characters REXX in the first line of the comment to help users identify the program as a REXX exec.

If your installation will primarily use REXX, it is recommended that you use SYSEXEC for your REXX execs. Using SYSEXEC makes it easier to maintain your REXX execs and is also useful for execs that you use on other SAA environments. If your installation primarily uses CLISTs and does not plan to have a large number of REXX execs, you can use SYSPROC for your CLISTs and REXX execs.

With TSO/E, the author of a REXX exec can specify whether the exec should be compressed. A REXX exec is eligible to be compressed when the REXX exec is implicitly invoked and loaded from either SYSPROC or an application-level CLIST library defined by ALTLIB.

One use of compression is to reduce the size of an exec stored in VLF. Be aware that even if the exec cannot be stored in VLF (for example, VLF is not active) the exec can still be compressed. For details on REXX compression, refer to *z/OS TSO/E Programming Guide* and *z/OS TSO/E REXX Reference*.

CLIST programming language

The CLIST language is a high-level interpretive language that enables you to work more efficiently with TSO/E. You can write programs called CLISTs to perform routine and complex programming tasks on TSO/E.

The term CLIST is short for *Command List* because the most basic CLISTs are lists of TSO/E commands. When you execute such a CLIST, it issues the commands in sequence. CLISTs can also perform more complex tasks such as displaying a series of full-screen panels and providing interfaces to applications written in other languages.

The CLIST language provides a wide range of programming functions. Its features include:

- CLIST statements that allow you to write structured programs, perform I/O, define and modify variables, and handle errors and attention interruptions.
- Arithmetic and logical operators for processing numerical data
- String-handling functions for processing character data.

Because the CLIST language is an interpretive language, CLISTs are easy to write and test and do not require you to compile or link-edit them. To test a CLIST, you simply execute it, correct any errors, and then re-execute it.

For complete information about writing, testing, and executing CLISTs, see *z/OS TSO/E CLISTs*.

ALTLIB command

When you implicitly invoke a CLIST or REXX exec, the EXEC command processor, by default, searches system-level libraries allocated to a system file (SYSPROC or SYSEXEC). You can use the ALTLIB command to specify alternative CLIST or REXX exec libraries for implicit execution. This command performs the same function for CLISTs and REXX execs as the LIBDEF command in ISPF performs for panels, messages, tables, skeletons, and load libraries. With ALTLIB, a user or ISPF application can easily activate and deactivate CLIST and REXX exec libraries as the need arises. This flexibility provides a potential performance benefit by reducing the time it takes to search for these libraries.

The ALTLIB command lets you specify alternative libraries on the user, application, and system-level. The user-level includes libraries previously allocated to the SYSUEXEC or SYSUPROC file. The application-level includes libraries specified on the ALTLIB command using the data set or file name. The system-level includes libraries previously allocated to the SYSEXEC or SYSPROC file.

During implicit execution, the EXEC command processor searches the libraries in the following order: user-level, application-level, and system-level. At the application and system-levels, the ALTLIB command uses the virtual lookaside facility (VLF) to provide potential increases in library search speed.

Using various operands on the ALTLIB command, a user or ISPF application can:

- Define alternative application-level libraries of CLISTs or REXX execs.
- Activate the search for execs in a library or libraries at the specified level(s), in the order specified.
- Exclude one or more library levels (user, application, system) from being searched.
- Reset the search order to the system level only.
- Obtain a display of the current search order.

For more information about using the ALTLIB command, see *z/OS TSO/E Command Reference*.

TSOLIB command

The TSOLIB command provides system and application programmers with a means of linking to different versions of load module libraries while remaining in their active TSO/E session. TSO/E searches these libraries before those specified in the user's logon procedures and those searched by default. This provides for reduced access time to the modules the programmer may be working on and allows a flexible way to change a user's environment.

With the TSOLIB command, a previous habit of having different user IDs or different logon procedures available for these users is no longer required.

Using various operands of the TSOLIB command users can:

- Activate and deactivate load module libraries of their choice.
- Stack activation requests. Thus, the latest activation of libraries replaces a previous activation until the latest one is deactivated again. This provides for effective setups if, for example, a programmer works on several different versions of a product.
- Reset the search order for load module libraries to its original state.
- Display the current search order and any stacked activations.

For more information about using the TSOLIB command, see *z/OS TSO/E Command Reference*.

Support for writing Command Processors

TSO/E provides commands that you can use to perform a wide variety of tasks. For example, you can use TSO/E commands to define and maintain data sets, and write and test programs.

You can write Command Processors to replace or add to this set of commands. By writing your own Command Processors, your installation can add to or modify TSO/E to better suit the needs of its users. For TSO/E, the IBM Compiler and Library for REXX/370 contains features to assist you in writing simple Command Processors.

A command processor is a program written in assembler language that receives control when a user at a terminal enters a command name. It receives control from the terminal monitor program (TMP), a program that provides an interface between terminal users and Command Processors, and has access to many system services.

The main difference between Command Processors and other programs is that when a command processor is invoked, it is passed a command processor parameter list (CPPL) that gives the program access to information about the caller and to system services.

Command processors must be able to communicate with the user at the terminal, as well as respond to abnormal terminations and attention interruptions. Command processors can recognize subcommand names entered by the terminal user and then load and pass control to the appropriate subcommand processor.

Command processors can use many of the TSO/E programming services described in the following section to perform required functions. For guidelines on how to write a command processor, what TSO/E services to use, and how to test and install the command processor, see *z/OS TSO/E Programming Guide*.

Support for using TSO/E services in non-TSO/E environment

TSO/E environment service allows applications to use a subset of TSO/E services and unauthorized commands without starting a TSO/E session. This supports applications that are not running under the terminal monitor program (TMP) IKJEFT01. It also includes writing transaction programs that can access TSO/E services from outside of the TMP.

A program can use the TSO/E environment service to build and initialize a TSO/E environment instead of using the TSO/E terminal monitor program (TMP). The environment consists of a set of initialized TSO/E control blocks, an initialized I/O stack, and settings to indicate background processing. A REXX language processor environment will also be initialized if a previous REXX environment does not exist.

There are some limitations to the TSO/E environment service because of the internal task structure necessary for some functions in TSO/E. These internal task structures cannot be created without the TMP. Commands and operations that are *not* supported include:

- Authorized commands
- FIB (Foreground Initiated Background) commands
- TEST command
- CONSOLE command
- Attention processing
- Session Manager operations (except SMCOPY)
- ALLOCATE command (using the ALTFILE keyword)

For complete information about how to use the TSO/E environment service, see *z/OS TSO/E Programming Services*.

Programming services

TSO/E provides many services that you can use in system or application programs to perform various tasks. These services consist of programs, macros, and CLISTs.

TSO/E services support a wide range of functions that are useful in writing system programs as well as application programs that exploit the full-screen capabilities of TSO/E.

The following table describes the tasks you can perform and the service(s) that support the task. For complete information about how to use these services, see *z/OS TSO/E Programming Services*.

Task	Service
Check the syntax of subcommand names	Command scan service routine
Check the syntax of command and subcommand operands	Parse service routine
Control terminal functions and attributes, such as full-screen mode and terminal line size	Terminal control macro instructions
Process terminal I/O and issue messages	• BSAM and QSAM macro instructions • TSO/E I/O service routines • TGET/TPUT/TPG • TSO/E message handling routine
Retrieve system messages that are issued during a console session	GETMSG service
Cause the system to recognize and schedule an attention exit that receives control when an attention interruption occurs	STAX service routine
Process a CLIST's attention routine when an attention interruption occurs	CLIST attention facility
Obtain a list of data set names that match specified criteria	ICQGCL00
Ensure that a specified data set has adequate free space for additional data	Space management CLIST
Create or delete alternative library environments and modify alternative library definitions for CLIST and REXX libraries	Alternative library interface routine

Task	Service
Allocate, free, concatenate, and deconcatenate data sets during program execution	Dynamic allocation interface routine
Retrieve information from the system catalog, such as data set name, index name, control volume address or volume ID	Catalog information routine
Construct a fully-qualified data set name when a partially-qualified name is entered by a terminal user	Default service routine
Analyze return codes from dynamic allocation or DAIR and issue appropriate error messages	DAIRFAIL routine
Analyze VSAM macro instruction failures, subsystem request failures, parse service routine or PUTLINE failures, and ABEND codes, and issue an appropriate error message	GNRLFAIL/VSAMFAIL routine
Determine if a command or program is authorized; determine if a command is not supported in the background	Table look-up service
Invoke commands, CLISTs, REXX execs and other programs **Note:** Unauthorized programs can use the TSO/E service facility to invoke APF-authorized functions.	TSO/E service facility
Establish a TSO/E environment outside of the TSO/E TMP	TSO/E environment service
Create, update, and return the values of CLIST and REXX variables	Variable access routine
Retrieve information from the Information Center Facility's names directory, such as phone numbers, user IDs, and addresses for specified names	ICQCAL00 program
Display lists of printers for users to select and to print data sets on selected printers	Printer support CLISTs
Invoke Information Center Facility applications	Application invocation function (ICQAMLIO)

TEST and TESTAUTH commands

The TSO/E TEST and TESTAUTH commands let you test assembler language programs, including command processors or other programs written in assembler. You can use the TEST command to test unauthorized programs including those that use access registers. You can use the TESTAUTH command to test APF-authorized programs. You can also test APPC/MVS transaction programs with the TEST and TESTAUTH commands. You must be defined to the RACF TESTAUTH resource to use TESTAUTH. For more information about the functions you can perform using the TEST and TESTAUTH commands, see Chapter 6, “Diagnosis,” on page 55.

Command package

The TSO/E Command Package is integrated into TSO/E. It includes:

- Support for running terminal sessions as batch jobs
- Automatic saving of data

- Accounting facilities enhancements
- Defaults for the user attribute data set
- ATTRIB and FREE subcommands
- ALL keyword for the FREE command and subcommand
- Eight-character station ID

Running terminal sessions as batch jobs

You can run a terminal session as a batch job. You might do this when it is impractical for you to enter commands from a terminal, as when your job has a long run time or produces large amounts of printed output.

Instead of waiting at a terminal for your job to run, you can use the terminal to prepare a job containing the commands and data you would have entered at the terminal. Then use the SUBMIT command to run the job. In this case, you are using the facilities of TSO/E exactly as if you had submitted the commands individually at the terminal.

For your job, you need these job control language (JCL) statements:

- A JOB statement to identify your job.
- An EXEC statement with the name of the TSO/E terminal monitor program (IKJEFT01, IKJEFT1A, or IKJEFT1B).
- Special DD statements to indicate that your input data contains executable TSO/E commands, and to indicate what you want done with your output.

You also need commands (such as LOGON and EDIT) that you would have entered at the terminal to run your job. You need the commands in the sequence you would have entered them.

If you create a data set and use the SUBMIT command, TSO/E provides a JOB statement for you. You need to provide only the commands you want executed.

Using the SUBMIT command

You can use the SUBMIT command without first creating a control data set that contains the job stream to be submitted. You can do this in one of the following ways:

- Enter a SUBMIT * command and then enter the job stream from the terminal.
- Put your JCL into a CLIST along with a SUBMIT * command and then invoke that CLIST.

The SUBMIT command has two additional operands:

- END(nn) where nn denotes the characters you use to end job streams containing blank lines (without the END operand, the first blank or null line ends your job stream).
- PAUSE stops processing after the last line of your job stream is read, so you can decide whether to continue with submission.

Automatic saving of data

When you are doing text editing, TSO/E can save your data automatically in a work file so you won't have to remember to type SAVE. If you finish editing without a problem, TSO/E deletes the work file. If a problem does occur, TSO/E makes the work file available to you the next time you log on. You tell TSO/E how often to save your data in terms of number of changes or number of lines you enter.

Accounting facilities enhancements

You can use a type 32 System Management Facilities (SMF) record for a total count of each TSO/E command issued and all subcommands of the EDIT, ACCOUNT, OUTPUT, and OPERATOR commands, by command and subcommand type.

Defaults for the user attribute data set

Each entry in the user attribute data set (SYS1.UADS) has these defaults:

- A MSGCLASS default for submitted jobs where HOLD or NOHOLD is specified on the SUBMIT command.
- A job class default for jobs submitted without a JOB statement.
- A SYSOUT class default for job output where no SYSOUT class is specified on the ALLOCATE command.

ATTRIB and FREE subcommands

The EDIT command has ATTRIB and FREE subcommands identical to the ATTRIB and FREE commands.

ALL keyword for the FREE command and subcommand

Both the FREE command and the FREE subcommand of the EDIT command have an ALL keyword you can use to free all dynamically allocated data sets.

Eight-character station ID

You can use an eight-character station ID with the DEST keyword of the ALLOCATE, FREE, and OUTPUT commands.

APPC/MVS administration dialog

The APPC/MVS administration dialog allows Information Center Facility administrators to perform interactive maintenance of the APPC/MVS system data files. The dialog supports functions provided by the APPC/MVS administration utility to maintain transaction program profiles, side information, and data base tokens residing in APPC/MVS system data files. For more information about APPC/MVS administration dialog, see *z/OS MVS Planning: APPC/MVS Management*.

TSO/E Support for a REXX Compiler

TSO/E provides a defined interface to support a REXX compiler, such as the IBM Compiler and Library for REXX/370. The interface is a compiler programming table that identifies the compiler runtime processor and the catalog information routines. The compiler runtime processor is the program that executes compiled REXX execs.

The compiler programming table contains an entry for each compiler runtime processor and the interface routines that it uses. These routines allow a compiler runtime processor to perform certain tasks related to REXX language processing. The use of the four catalog information routines is optional. The interface routines are:

- Compiler Interface Initialization Routine

  If you install this routine, it receives control to initialize a compiler runtime processor before the compiler runtime processor is invoked the first time. The compiler interface initialization routine receives control once for each compiler runtime processor in a REXX language processor environment.
- TSO/E Diagnosis: TMP and Service Routines

  If you install this routine, it can be used to free resources that the compiler interface initialization routine and the compiler runtime processor might have obtained. The compiler interface termination routine receives control when the language processor environment is terminated.
- Compiler Interface Variable Handling Routine

  If you install this routine, it accesses variables of a compiled exec. The compiler interface variable handling routine receives control whenever an external routine or host command requests access to REXX variables using IKJCT441 or IRXEXCOM.

- Compiler Interface Load Routine

  If you install this routine, it loads a compiled REXX exec from storage or frees the in-storage control block. The compiler interface load routine receives control when TSO/E REXX loads a compiled exec into storage and when TSO/E REXX needs to free the in-storage control block of the compiled REXX exec.

Chapter 4. Customization

Customization is the process of tailoring TSO/E functions to fit the needs of your installation and making other IBM products available to TSO/E users. The following topics discuss a number of facilities to enable customization. For complete information, see *z/OS TSO/E Customization*.

TSO/E environment

Customization of the TSO/E environment generally refers to customization that makes a TSO/E facility available or customization that changes default values that affect TSO/E. Customizing the TSO/E environment includes customizing:

- VTAM® and TCAM
- Logon limits
- The logon/logoff process
- ISPF/PDF
- Authorized commands and programs and commands not supported in the background
- TRANSMIT and RECEIVE availability
- HELP data set usage
- Language enablement support
- Command/program invocation platform support

You can also customize the TSO/E environment by providing TSO/E resource protection.

VTAM and TCAM

Before users can log on to TSO/E, you must define TSO/E to one of two access methods:

- Advanced Communications Function for VTAM [2]
- Advanced Communications Function for TCAM [3]

You can then customize how TSO/E works with either VTAM or TCAM to suit the needs of your installation. SYS1.PARMLIB has members that contain default values used to initialize TSO/VTAM or TSO/TCAM. Both VTAM and TCAM installations can change or override these values to customize the TSO/E environment.

VTAM installations can also tailor the session protocols that VTAM uses to start a session between a terminal and TSO/E. Session protocols define the rules that VTAM uses to manage a terminal session. If you wish, you can modify the protocols for all users or TSO/E users only.

Some terminal keyboards may not contain all the characters that a VTAM installation needs. If this is true, you can provide translation tables that allow TSO/VTAM users to internally replace (or represent) unavailable characters with characters that are available on the keyboard.

Logon limits

TSO/E provides ways to limit and manage the maximum number of concurrent logons, limit the user's region size, limit user access to applications, and limit which systems a user can access.

You can change the maximum number of users that can be logged on to TSO/E concurrently by adjusting the factors that determine the maximum number. If necessary, you can also write a logon pre-prompt exit routine to allow or disallow a user to log on depending on the user group to which the individual is assigned.

[2] Virtual Telecommunications Access Method
[3] Telecommunications Access Method

You can define a region size default that applies to all users, that varies from one logon procedure to another, or that varies from one user to another. You can also use the logon pre-prompt exit to monitor and adjust the region size requested by a user.

You can limit the applications to which a user has access by providing several logon procedures, each providing access to different applications, and by restricting users to only certain logon procedures.

You can limit which systems a user can access in a sysplex using the APPL class in RACF and the LOGON VERIFYAPPL(ON) parmlib option for TSO/E.

The Logon/Logoff process

TSO/E provides several ways to customize the logon/logoff process. By changing initialization values, you can:

- Change how often the system displays the `logon proceeding` message.
- Limit the number of attempts a user can make at entering information in response to log on prompts.
- Tailor the reconnect option.
- Suppress messages that are generated during the execution of the logon job.
- Allow password phrases up to 100 characters in length.

If the LOGON PASSPHRASE(ON) parmlib setting is active, the full screen LOGON panel prompts for passwords in a 100–character PASSWORD field. It does not display the NEW PASSWORD field, but has a new option to select at the bottom of the panel. If an "S" is entered for that option, a prompt for a new password is issued which reuses the PASSWORD field.

You can also review and adjust factors that affect logon performance, such as using STEPLIBs in logon JCL, and you can write exits to further customize the logon/logoff process.

TSO/E provides authorized and unauthorized logon pre-prompt exits that allow monitoring, changing, or supplementing information that is provided on the LOGON command, cancel a logon, or interact with the user by sending messages and requesting a reply. Additional *authorized* functions can also be performed because of the authorized log on pre-promt exit. If you use the authorized exit, you can:

- Specify the first TSO/E user command to be issued in the session.
- Return job and SYSOUT classes.
- Bypass RACF processing.
- Specify the relative block address (RBA) of the user's mail directory.
- Provide a four-byte user word of information to be used during the session or at logoff.
- Specify a security label (SECLABEL) to be used for the session. (SECLABEL is recognized only if RACF is installed and security label checking has been activated.) For more information about security labels and the logon process, see “Using security labels” on page 31.
- Specify languages to be used in displaying translated information.
- Set up a console profile.

TSO/E provides a post-prompt exit that allows further customizing the logon process. After the prompting for logon input is complete, you can use this exit to add, examine, and modify JCL statements associated with logon.

TSO/E allows customizing logon panels and logon help text panels using logon panel modules. If a language feature is installed, a logon panel for that language is available. There is one logon panel module for each language.

You can also use the pre-display and post-display exits to customize the logon process. These exits allow supplying default information, update information, validate user-supplied information, and reprompt the user for information, if needed. Using these exits that can be performed, but are not limited to, the following functions:

The pre-display exit allows (1) updating information that is contained on the logon panel, and (2) process any installation-defined fields on the logon panel.

The post-display exit allows you to (1) processing and validate fields on the logon panel, and (2) request display of help screens.

TSO/E also provides an *authorized* logoff exit. Use the logoff exit to perform clean-up operations, gather accounting information, control information that is written to the UADS and RACF database, and issue the LOGOFF or LOGON command to control relogons.

Using security labels

If your installation has RACF active, you can define security labels (SECLABELs) for users and activate security label checking. The LOGON command and full-screen logon panel allow users to specify a security label to be associated with their session. If security label checking is active, all logon attempts will be audited. For more information about setting up SECLABELs and activating security label checking, see *z/OS Security Server RACF Security Administrator's Guide*.

ISPF/PDF

The Interactive System Productivity Facility/Program Development Facility (ISPF/PDF) helps users develop different types of applications, including dialogs. It uses display terminals in an interactive environment to assist with many programming tasks.

You can enable TSO/E users to use ISPF/PDF, by defining ISPF/PDF to TSO/E. You do this by modifying the users' logon procedures to allocate the ISPF/PDF data sets.

In addition, you can modify the ISPF default list of TSO/E commands that a user can issue from ISPF/PDF panels and you can allow users to use the Session Manager from ISPF/PDF.

Authorized commands and programs and commands not supported in the background

TSO/E users cannot use authorized commands or programs until you make them available by defining them in certain SYS1.PARMLIB members or CSECTs. TSO/E requires you to specify certain authorized commands and programs that users are able to use and certain commands that are not supported in the background. You can optionally add authorized commands and programs, such as VLFNOTE, LISTDS, and IEHMOVE, to the list to make them available to users at your installation. You can also add commands that you do not want users to execute in background jobs, such as user-written commands that do not work properly in the background.

You can use the PARMLIB command to dynamically make the commands or programs in SYS1.PARMLIB member IKJTSOxx available or restricted, without having to re-IPL the system. You can also use the PARMLIB command to view the current list of authorized commands and programs, and commands not supported in the background. For more information about using SYS1.PARMLIB and the PARMLIB command, see “SYS1.PARMLIB data set” on page 37.

TSO/E provides a table look-up service that also helps simplify installation. This service allows you to determine whether a given program or command is defined as authorized and whether a given command is not available for use in the background.

TRANSMIT and RECEIVE availability

To enable TSO/E users to use the TRANSMIT and RECEIVE commands, you must add these commands to the list of authorized commands and specify installation defaults to control TRANSMIT and RECEIVE processing. You may also need to modify certain JES initialization statements for the commands to function properly.

You can use the IKJTSOxx member of SYS1.PARMLIB to specify TRANSMIT and RECEIVE installation defaults. You can also use the PARMLIB command to list the current defaults and dynamically update them (with the contents of IKJTSOxx) without having to re-IPL the system. For more information about using SYS1.PARMLIB and the PARMLIB command, see “SYS1.PARMLIB data set” on page 37.

HELP data set usage

By default, TSO/E provides the prompt mode HELP function for several commands. The prompt mode HELP function provides help for a user who omits or incorrectly specifies a positional operand on a TSO/E command. You can customize HELP processing by providing the prompt mode HELP function for additional commands. You can also set up HELP data set members so they merge information from other members for display to the user.

Language enablement support

TSO/E takes advantage of the MVS message service to allow installations to provide TSO/E information to users in languages other than English, including languages that require a double-byte character set. TSO/E information includes TSO/E messages, help information, the TRANSMIT full-screen panel and logon panels. The CONSOLE command also supports the display of translated system messages that are issued during a console session.

To provide translated information, you need to initialize and activate the MVS message service and set up languages for users. For help information, you also need to specify help data sets for each language.

To initialize and activate the MVS message service, see *z/OS MVS Planning: Operations*.

To set up languages for users, you can use the following methods. If your installation has RACF, you can use RACF commands to set up languages for users. You can also use the authorized logon pre-prompt exit IKJEFLD1 to set up languages. Individual users can set up or change languages for themselves using the PROFILE command.

To provide translated help information, you must specify the help data sets to be used for different languages. You can use the IKJTSOxx member of SYS1.PARMLIB to define these data sets. In IKJTSOxx, you can (1) specify help for any number of languages, and (2) specify up to 255 data sets to be searched for help text in a particular language.

You can use the PARMLIB command to list the current HELP defaults in SYS1.PARMLIB and dynamically update them (with the contents of IKJTSOxx) without having to re-IPL the system. For more information about using SYS1.PARMLIB and the PARMLIB command, see “SYS1.PARMLIB data set” on page 37.

Note: Language translation does not occur with Session Manager.

TSO/E allows you to customize the logon panels and the logon help panels. If a language feature is installed, a logon panel for that language is available. You can also use the pre-display and post-display exits to customize the logon process. For more information on these exits, see “The Logon/Logoff process” on page 30.

Command/program invocation platform support

The TSO/E service facility allows installations to specify commands and programs that are to run on a command/program invocation platform. This platform helps to reduce the system overhead associated with the initialization and termination of commands and programs.

You can identify which commands and programs are allowed to execute on a command/program invocation platform by using the PLATPGM and PLATCMD statements in SYS1.PARMLIB member IKJTSOxx.

The TSO/E service facility gives you control about the initialization and termination of the command/program invocation platform environment.

For more information about these enhancements, see *z/OS TSO/E Programming Services*.

TSO/E resource protection

You can protect TSO/E resources by limiting the commands that users can issue and by limiting user access to data sets. You can limit the commands users can issue from TSO/E READY mode, from Session Manager, from the background, and from ISPF/PDF.

By default, users cannot use the ACCOUNT, CANCEL, CONSOLE, CONSPROF, OPERATOR, OUTPUT, PARMLIB, RACONVRT, STATUS, SUBMIT, SYNC, and TESTAUTH commands from TSO/E READY mode. You can give users authority to use these commands when you define the users to TSO/E using the ACCOUNT command and/or RACF commands. You can optionally write TSO/E exits to authorize users to use the CONSOLE, CONSPROF, PARMLIB, and TESTAUTH commands. For the CANCEL, OUTPUT, STATUS, and SUBMIT commands, you can also write TSO/E exits and exits provided by JES2 and JES3 to customize and restrict how users submit jobs and process job output.

By default, users can issue the EXEC, FREE, LISTBC, OUTDES, PRINTDS, RECEIVE, SEND, and TRANSMIT commands from TSO/E READY mode. You can write TSO/E exits to restrict the use of these commands.

For the OPERATOR command, you can write exits for the SEND subcommand to restrict users who are authorized to use the OPERATOR command from using the SEND subcommand.

By default, a user using Session Manager can issue all TSO/E commands and Session Manager commands. You can limit the commands a user can issue from Session Manager by writing Session Manager exits.

Users, by default, can also issue most TSO/E commands from the background and from ISPF/PDF panels. You can limit the use of commands in the background by changing a SYS1.PARMLIB member or by coding a TSO/E CSECT. You can make modifications to an ISPF/PDF module to limit the commands used from ISPF/PDF.

You also have the option to limit a user's access to data sets. Both RACF and the MVS allocation input validation routine (IEFDB401) provide this capability.

With RACF active, you can use security labels (SECLABELs) and your security administrator can activate security label checking. In this case, resources and users have security labels associated with them. Users can access only those resources that they have been authorized to use through RACF.

Security label checking affects the processing of several TSO/E commands, such as SEND, LISTBC, TRANSMIT, and RECEIVE. For information about the processing of these commands with security labels, see *z/OS TSO/E Customization*.

For information about setting up security labels, see *z/OS Security Server RACF Security Administrator's Guide*.

The UADS and RACF data base

With RACF active, you have the option to define users through the RACF data base instead of through the UADS. Defining users through the RACF data base allows you to define users to TSO/E and RACF at the same time and eliminates the need to maintain two data sets for user information. Various RACF commands allow you to define users, maintain those definitions in the RACF data base, and remove users. For details about the RACF commands that define interactive users to TSO/E, see *z/OS Security Server RACF Command Language Reference*.

If your installation currently uses the UADS data set, you must first convert the UADS information to the RACF data base. Two TSO/E commands, RACONVRT and SYNC, assist installations in performing this conversion. The RACONVRT command converts user information to a format acceptable to RACF. The SYNC command or the SYNC subcommand of ACCOUNT synchronizes the RACF data base with the broadcast data set. For the syntax of the RACONVRT and SYNC commands, see *z/OS TSO/E System Programming Command Reference*.

TSO/E commands

TSO/E enables you to customize the use of many commands to suit your installation's data processing requirements. One way to customize a command is to write an exit routine. TSO/E provides exits for the following commands:

- ALLOCATE
- ALTLIB

- CANCEL
- CONSOLE
- CONSPROF
- EDIT - the COPY, MOVE, and RENUM subcommands
- EXEC
- FREE
- LISTBC
- LOGON and LOGOFF
- OPERATOR - the SEND subcommand
- OUTDES
- OUTPUT
- PARMLIB
- PRINTDS
- RECEIVE
- SEND
- STATUS
- SUBMIT
- TEST
- TESTAUTH
- TRANSMIT
- TSOLIB.

Using the exits, you can customize commands in various ways. At a minimum, you can use the exits to restrict users from using a command or to change the operands a user specifies.

In addition to exits, TSO/E provides other ways for you to customize certain commands. The following sections describe additional ways you can customize specific commands.

ALLOCATE command

The TSO/E ALLOCATE command lets users dynamically allocate and manage data sets and z/OS UNIX files. In addition to writing exit routines for the ALLOCATE command, you can:

- Use the MVS allocation input validation routine to monitor and, if necessary, change information that a user provides on the ALLOCATE command.
- Use Storage Management Subsystem (SMS) to manage your system's data set and storage and to simplify how users allocate data sets.
- Use output descriptors to eliminate the need for users to specify output-related operands on the ALLOCATE command. You can define the output descriptors using OUTPUT JCL statements in the user's logon procedure. JES installations can also define and reuse dynamic output descriptors using the TSO/E OUTDES command.
- Change the default parameter values used by the Information Center Facility space management service to manage data set space.

You can use member IKJTSOxx of SYS1.PARMLIB to set a default value (SHR or OLD) for the data set disposition specified on the ALLOCATE command. If a user issues the ALLOCATE command without specifying a data set disposition, the disposition defaults to the setting in IKJTSOxx.

You can make the default data set disposition in IKJTSOxx take effect immediately using the PARMLIB command with the UPDATE operand. You can also list the current ALLOCATE default using the PARMLIB command with the LIST operand.

For more information about using SYS1.PARMLIB and the PARMLIB command see "SYS1.PARMLIB data set" on page 37.

CANCEL, OUTPUT, STATUS, and SUBMIT commands

The SUBMIT command allows users to submit jobs for processing by JES. Users can issue the OUTPUT command to process the output of jobs they submit and the CANCEL command to stop the processing of jobs they submit. Users can also issue the STATUS command to display the status of any job in the system.

In addition to writing TSO/E exit routines for the CANCEL, OUTPUT, STATUS and SUBMIT commands, you can:

- Use JES initialization statements to change the default processing performed for jobs submitted by TSO/E users
- Use JES exits and SMF exits to perform similar processing as the TSO/E SUBMIT exit and the TSO/E OUTPUT, STATUS, CANCEL exit
- Use RACF resource classes to customize the way TSO/E users submit jobs and process the output.

CONSOLE and CONSPROF commands

The CONSOLE command allows users with CONSOLE command authority to perform MVS operator functions from a TSO/E terminal. The CONSOLE command establishes an extended MCS console session with MVS console services. During an extended MCS console session, users can enter MVS system and subsystem commands and obtain responses to those commands.

The CONSPROF command allows users with CONSOLE command authority to modify the console profile to tailor message processing during an extended MCS console session.

In addition to writing TSO/E exit routines for the CONSOLE and CONSPROF commands, you can:

- Use the logon authorized pre-prompt exit IKJEFLD1 to set up a console profile for users.
- Use the IKJTSOxx member of SYS1.PARMLIB to define message processing defaults for the CONSOLE command.

You can use the PARMLIB command to list the current CONSOLE command defaults and dynamically update them (with the contents of IKJTSOxx) without having to re-IPL the system. For more information about using SYS1.PARMLIB and the PARMLIB command, see "SYS1.PARMLIB data set" on page 37.

You can also specify console attributes for each TSO/E user having CONSOLE command authority. The console attributes control various functions, including the types of MVS commands a user can issue during an extended MCS console session, the routing of MVS messages and commands, and the display of MVS message formats. If your installation has RACF active, you can optionally define an OPERPARM segment with the console attributes in the user's RACF profile. If an OPERPARM segment has been defined for a user, the user's console attributes are saved from session to session. You can use RACF ADDUSER and ALTUSER commands to define OPERPARM segments for users.

If you do not specify console attributes for a user, the system defaults are used. For more information about each of the console attributes and their defaults, see *z/OS MVS Planning: Operations*.

RACF resource classes

If your installation has RACF installed, you can use the RACF resource classes JESSPOOL, JESJOBS, and SURROGAT to customize job and output processing. You can use the JESSPOOL resource class to protect against unauthorized spool access of the SYSOUT data sets for the TSO/E OUTPUT command. The JESJOBS resource class lets you control who can submit and cancel jobs by job name and the SURROGAT resource class provides the capability for surrogate job submission. If you define users to the RACF SURROGAT resource class, jobs submitted by surrogate users can be cancelled and/or viewed by surrogate users without knowing the user's password.

For information about using the RACF resource classes JESSPOOL, JESJOBS, and SURROGAT, see *z/OS Security Server RACF Security Administrator's Guide*. For information about submitting a job as a surrogate user, see *z/OS Security Server RACF General User's Guide*.

EDIT command

The TSO/E EDIT command with its subcommands allow users to create, modify, store, submit, retrieve, and delete data sets with sequential or partitioned data set organization. To suit the needs of your installation, you can customize the EDIT command by writing exit routines and using the facilities described below:

- TSO/E has defined a number of data set types. If these types do not meet your installation's needs, you can either change the attributes of the TSO/E defined data set types or you can define your own data set types.
- For each installation-defined data set type, you can write a syntax checker and a syntax checker exit. The syntax checker can detect errors when a user edits a data set of the type recognized by the syntax checker. You can use the exit to obtain information that the user specifies on the EDIT command.
- If the functions provided by the EDIT command do not meet your installation's needs, you can supplement those functions by writing your own sub Command Processors and adding them to the system.
- The EDIT command requires the allocation of data sets for work space. You have the option to either preallocate utility work data sets, or to allow EDIT to allocate them dynamically as space is needed.

LISTBC, OPERATOR SEND, and SEND commands

The SEND command allows users to send messages to other users. Users who are authorized to use the OPERATOR command can issue the SEND subcommand to send messages to users. Depending on the operands the user specifies, the SEND command and OPERATOR SEND subcommand may store messages in the broadcast data set or individual user logs that you define. Users issue the LISTBC command to retrieve commands that SEND stores. In addition to writing exit routines, you can customize the SEND, OPERATOR SEND, and LISTBC commands by using the IKJTSOxx member in SYS1.PARMLIB to define installation defaults. In IKJTSOxx, you can:

- Specify whether users can use the SEND command
- Specify whether users authorized to use the OPERATOR command can use the SEND subcommand
- Specify whether user logs are to be used and if they are, name those logs
- Specify whether the SEND command stores messages in the broadcast data set, if your installation is using user logs, and the target user does not have a user log
- Specify whether the individual user logs are security protected from the user
- Specify the broadcast data set using the BROADCAST parameter

You can use the PARMLIB command to list the current SEND command processor defaults in SYS1.PARMLIB and dynamically update them (with the contents of IKJTSOxx) without having to re-IPL the system.

For more information about using SYS1.PARMLIB and the PARMLIB command, see “SYS1.PARMLIB data set” on page 37.

With TSO/E, and RACF installed, your installation can use the RACF security resource message class SMESSAGE to control which users can send messages to other users. For example, a user may send messages to another user only if permitted to the receiving user's resource within the SMESSAGE class. If a user does not have a resource defined in the SMESSAGE class, any user can send a message to that user. For information about setting up the SMESSAGE resource class, see *z/OS Security Server RACF Security Administrator's Guide*.

When you use the PARMLIB command to update system defaults, the SEND statement's SYSPLEXSHR setting is updated on all systems in the sysplex. Sysplex communication facilities are used to notify the

other systems of the change. As a result, a new message appears when the PARMLIB command lists the default settings.

If your installation shares the broadcast data set, you should ensure that you set the SYSPLEXSHR keyword correctly. For more information on setting the SYSPLEXSHR keyword, see *z/OS TSO/E Customization*.

PRINTDS command

The PRINTDS command allows users to print data sets. TSO/E provides two exits that enable you to control the use of the PRINTDS command. In addition, you can use output descriptors to simplify the use of the PRINTDS command. Output descriptors eliminate the need for users to specify output-related operands on the PRINTDS command. You can define the output descriptors using OUTPUT JCL statements in the user's logon procedure. JES2 installations can also define and reuse dynamic output descriptors using the TSO/E OUTDES command.

TEST and TESTAUTH commands

The TEST command allows users to test unauthorized assembler programs. The TESTAUTH command enables users to test APF-authorized assembler programs. Users can also test APPC/MVS transaction programs written in assembler language with the TEST and TESTAUTH commands.

You can customize the TEST and TESTAUTH commands by supplying installation-written subcommands and installation-written Command Processors to be invoked under TEST and TESTAUTH. You must define the subcommands and Command Processors to the TEST command using either (1) CSECT IKJEGSCU, or (2) SYS1.PARMLIB member IKJTSOxx.

You can use the PARMLIB command to list the current system defaults for the installation-written TEST subcommands and the TSO/E commands that run under TEST. You can also use the PARMLIB command to dynamically update these defaults (with the contents of IKJTSOxx) without having to re-IPL the system.

For more information about using SYS1.PARMLIB and the PARMLIB command, see “SYS1.PARMLIB data set” on page 37.

TRANSMIT and RECEIVE commands

The TRANSMIT command allow users to transmit messages and data sets. The RECEIVE command allows users to receive messages and data sets sent by the TRANSMIT command. You can protect the security classification of messages and data sets if your installation has RACF active. You can provide this protection by using security labels (SECLABELs) and activating security label checking. When security label checking is active, data sets and messages have security labels associated with them and security checks are performed on all transmitted information. The security label of the receiving user's logon session must be equal to or greater than the security label of the transmitted information for the user to receive the message. For more information about TRANSMIT/RECEIVE processing when security label checking is active, see *z/OS TSO/E Customization*.

The TRANSMIT and RECEIVE commands provide adherence to distributed data management (DDM) architecture. Data sets (sequential and PDSEs) that are transmitted and received by TSO/E will retain their explicitly defined DDM attributes.

SYS1.PARMLIB data set

You can simplify the installation process by specifying system defaults in members (IKJTSOxx) of the SYS1.PARMLIB data set instead of in tables. Using SYS1.PARMLIB members makes it easier to customize a system because installations need to change only one file. If you use tables, you must save them from one release to the next, and you must reassemble and link-edit them every time you make updates.

You can use SYS1.PARMLIB member IKJTSOxx to specify TSO/E system defaults for:

- ALLOCATE, CONSOLE, HELP, RECEIVE, SEND, TRANSMIT commands

- Installation-written TEST subcommands
- TSO/E commands that run under the TEST command
- Authorized commands and programs
- Commands that users cannot run in the background
- Commands and programs eligible for command or program invocation platform processing
- The broadcast data set name because SYS1.BRODCAST is no longer the only broadcast data set name allowed
- The password because up to 255 bytes are allowed to set for the LOGON password

To simplify the creation of the PARMLIB members, SYS1.SAMPLIB contains samples that a system programmer can copy to SYS1.PARMLIB and then change to suit the needs of the installation. For example, the IKJTSO00 member of SYS1.SAMPLIB contains samples of the statements you can use in the IKJTSOxx member. The changes to SYS1.PARMLIB take effect at the next IPL, or when you use the PARMLIB command.

Setting and displaying PARMLIB data

You can use the PARMLIB command to do the following:

- Dynamically UPDATE TSO/E system defaults (with the contents of IKJTSOxx)
- Switch to a new broadcast data set, without having to reIPL the system
- LIST current TSO/E system defaults and the name and volume of the current broadcast data set
- CHECK the syntax of any IKJTSOxx member of SYS1.PARMLIB
- Verify the LOGON password that can be up to 255 bytes

Note: The entry for the broadcast data set is no longer included in master JCL. The IKJTSOxx parmlib member can be specified on the IPL parameters. TSO/E allocates the broadcast data set during IPL.

For complete information about using the PARMLIB command, see *z/OS TSO/E System Programming Command Reference*. For information about creating and initializing IKJTSOxx members of SYS1.PARMLIB, see *z/OS MVS Initialization and Tuning Guide*.

You can use the SET IKJTSO=xx system command to do the following:

- Dynamically UPDATE TSO/E system defaults (with the contents of IKJTSOxx).
- Dynamically switch to a new broadcast data set, without having to reIPL the system.

For information about the SET IKJTSO=xx command, see *z/OS MVS System Commands*.

You can display the specifications in the active IKJTSOxx parmlib member using the DISPLAY IKJTSO system command. For information about the DISPLAY IKJTSO command, see *z/OS MVS System Commands*.

CLIST processing

TSO/E provides two exits you can use to customize CLIST processing. These exits enable you to define and process your own built-in functions and CLIST statements, and do your own processing of TSO/E commands.

REXX processing

TSO/E provides several services that let you customize REXX processing by tailoring the environment within which a REXX exec is interpreted. Many of these services let you change how an exec is processed and how the language processor interfaces with the system to access and use system services, such as I/O. Customization services include the following:

- Language processor environment characteristics

- Replaceable service routines
- REXX exits.

For complete information about changing environment characteristics and using replaceable routines, see *z/OS TSO/E REXX Reference*. For information about using the REXX exits, see *z/OS TSO/E Customization*.

Language Processor Environment characteristics

TSO/E provides default values for initializing Language Processor Environments for TSO/E (READY mode), ISPF, and non-TSO/E address spaces. Language Processor Environments define various characteristics relating to how execs are processed and how system services are accessed and used. You can change the values that TSO/E uses to define Language Processor Environments. You can also create your own environments at any time in any address space.

Replaceable service routines

When a REXX exec executes, various system services are used to perform I/O, load and free execs, obtain and free storage, and handle data stack requests. TSO/E provides several routines to handle these types of system services. These routines are known as *replaceable routines* because you can replace the system routines with your own routines. Your routines can check the request for a system service, change the request if needed, and then call the system-supplied routine to actually perform the service. Optionally, your routine can perform the request and not call the system-supplied routine.

You can provide your own routines in non-TSO/E address spaces and in language processor environments that are not integrated into TSO/E.

REXX exits

TSO/E provides several exits that you can use to customize REXX processing on a language processor environment basis. These exits enable you to perform various functions such as:

- Prevent the initialization of a language processor environment, change parameters used to initialize a language processor environment, or perform special pre-environment processing.
- Perform special attention processing in the TSO/E address space.

Session Manager

The Session Manager is an interface to TSO/E. It saves the commands entered by a user and the responses to the commands. It also allows the user to redisplay and change the commands or print them.

You can customize the use of Session Manager by either modifying the environment definition provided by IBM or supplying your own environment definition(s). The environment definition defines the user's screen layout, program function key definitions, and stream defaults. You can provide multiple definitions so different users can use different definitions. Multiple definitions allow you to customize the environment definition to individual user needs.

Three exits allow you to further customize Session Manager. The Session Manager exits enable you to monitor a user's interaction with the system while they are using Session Manager. You can use these exits to specifically:

- Monitor and intercept certain commands.
- Retain a log of a user's TSO/E session.
- Determine how long it takes a command to execute.
- Determine at what time certain operations were performed.
- Provide multiple default environments.

Information Center Facility

The Information Center Facility allows users to access products and services through a series of conversational panel dialogs. You can customize the Information Center Facility by:

- Adding, modifying, or deleting a product or service.
- Creating or tailoring application definitions for user groups, departments, and individual users.
- Adding commands to the command table that users can enter on the command line of a panel.
- Modifying Information Center Facility start-up and termination processing.
- Writing exits.

Adding, modifying, or deleting a product or service

You can add to the Information Center Facility any product or service that runs under the ISPF dialog manager and TSO/E. To add a product or service, you can use (1) Application Manager panels, and (2) the mass installation file process. You can also use Application Manager panels to modify or delete a product or service.

For complete information about adding, modifying, and deleting products and services to the Information Center Facility, see *z/OS TSO/E Administration*.

Adding a product or service

Adding a product or service involves defining the applications associated with it. There are three types of applications: panels, functions, and environments. The panels make up the interface that allows users to select different services, tutorials, and HELP information. Function, as it is used here, refers to the method used to invoke the panels. (For example, a CLIST, REXX exec, command, program, and menu are functions). An environment consists of a set of support information, such as commands for setup, invocation, and termination; data set allocations for panels, messages, tables, and load modules; and ISPF shared variables.

Using Application Manager panels, the Information Center Facility administrator can *add* a product or service by either:

- Entering information on various Application Manager panels to define the environments, functions, and panels associated with the product or service.
- Loading installation files that a system programmer created to define the product or service.

With TSO/E, panel and function installation files allow an INVOKING_PANEL entry, which integrates the application with existing panels. The INVOKING_PANEL entry allows you to specify the panel on which the application will be an option, the character(s) used to select the option, and the location on the panel where the option should appear.

For examples of how to create installation files, see *z/OS TSO/E Customization*.

With TSO/E, the Information Center Facility administrator can optionally use the mass installation file process to add products and services. An administrator can process multiple installation files at one time using a batch method. As each installation file is successfully processed, the installed application is verified and marked available in the Application Manager tables. For more information about the functions an administrator can perform using the mass installation file process, see “Applications” on page 51.

TSO/E also provides an upgrade function that lets administrators distribute updates to application definitions that are used across several locations and that may have local modifications. Instead of sending a complete installation file, administrators can send an upgrade file containing only the changes. Using an upgrade file minimizes the possibility of the changes conflicting with any local modifications.

Modifying or deleting a product or service

An Information Center Facility administrator uses the Application Manager panels to *modify* a product or service defined to the Information Center Facility or to *delete* it. As an alternative to deleting a product or service, the administrator can make it unavailable to users.

An administrator can optionally limit the availability of certain products or services to selected users or groups of users. See “Creating or tailoring application definitions” on page 41 for more information.

Creating or tailoring application definitions

Application Manager supports three levels of application definitions: system, group, and private. This support allows departments or user groups, and individual users to create or tailor panels, functions, and environments for their own use. The applications defined at the group and private levels can be either unique applications or copies of higher-level applications. At invocation time, lower-level application definitions that are actually copies will override the higher-level application definitions.

You can use this support to limit access to selected applications to private users or group(s) of users. For example, if you do not want all users to have access to APL2, you can copy down the applications that define APL2 to the group level or private level and then delete the system level APL2 application definitions.

You can also customize the Information Center Facility by limiting the use of group and private Application Manager. Users access group and private Application Manager off the PROGRAMMER SERVICES panel. If your installation does not want these services to be available to all users, you can copy down the applications that define these services to the group level and then delete the group and private Application Manager options from the system level panel.

Adding commands

To extend the function of the Information Center Facility, you can add your own commands to the command table, ICQCMDS. For example, you can add commands to enter dialogs, act as aliases of other commands, or to execute CLISTs. To use a command in the command table, a user simply types that command after the COMMAND or OPTION arrow on any panel in the Information Center Facility and presses the Enter key.

As shipped, the command table contains three commands. These commands allow you to quickly access, from any Information Center Facility panel, the user's or administrator's main menu panel, or a specific option (for example, the ENROLL option):

- The IC command displays the user's primary panel.
- The ADMIN command displays the administrator's primary panel.
- The GO commands goes to the option specified.

Modifying start-up and termination processing

An installation can easily tailor start-up and termination processing by modifying the CLISTs and applications that perform the processing.

The CLIST invocation statement defines how the Information Center Facility is to be started. You can modify start-up processing by changing the default parameter values on the CLIST invocation statement. For example, an installation can specify the first application to be invoked and the command string used to initialize the first application.

You can also modify start-up processing by providing a start-up function in the first application that the start-up CLIST invokes. The start-up function is performed before the processing for the application.

The termination CLIST contains no trace processing. If your installation requires trace processing, you can add the necessary code to the termination CLIST and then specify on the CLIST invocation statement that the termination CLIST is to be invoked when ISPF terminates.

See *z/OS TSO/E Customization* for more information about modifying Information Center Facility start-up and termination processing.

Writing exits

TSO/E provides several exits that you can use to customize the use of the Information Center Facility. These exits allow you to:

- Keep track of changes that users or administrators make in private and master names directories.
- Customize functions and panels that are invoked by the Application Manager.

If your installation has installed A Departmental Reporting System (ADRS) for use in the Information Center Facility, you can also write an exit routine for ADRS to perform various functions.

TSO/E exits

TSO/E provides exit points for many TSO/E functions and commands. At an exit point, the function or command invokes an exit routine if one exists. You can write an exit routine to perform special processing and customize how the function or command works. When your exit is finished processing, control returns to the command or function.

Exits allow you to change default values or extend a TSO/E function or command. The following table lists each function and command that you can write an exit for, the exit(s) provided in TSO/E for the function or command, and the possible uses for each exit. If the name of an exit must follow a naming convention, the table also shows what the name must be. For complete information about each exit, see the associated processing function in *z/OS TSO/E Customization*.

Table 2. Overview of Exit Points that TSO/E Provides

Processing	Exit	Uses of Exit
ALLOCATE command	Initialization IKJEFD47	Check and change the command users issue or provide pseudo-operands.
	Termination IKJEFD49	Perform clean-up processing. Specify an alternative return code.
ALTLIB command	Initialization IKJADINI	Change the command the user issues.
	Termination IKJADTER	Perform clean-up processing.
Application Manager	Function pre-initialization ICQAMFX1	Check a user's authorization to use an application, allocate data sets for an application, and prepare to gather accounting data.
	Function post-termination ICQAMFX2	Free data sets the function pre-initialization exit allocated and summarize accounting data.
	Panel pre-display ICQAMPX1	Set default values for the panel to be displayed.
	Panel post-display ICQAMPX2	Validate the information the user entered on the panel.
CANCEL	See the entry for the OUTPUT command in this table.	

Table 2. Overview of Exit Points that TSO/E Provides (continued)

Processing	Exit	Uses of Exit
CLIST processing	Built-in functions IKJCT44B	Add installation-written built-in functions.
	Statements IKJCT44S	Add installation-written CLIST statements.
CONSOLE command	Pre-parse IKJCNXPP	Check and, if necessary, change the command the user issues.
	Activation IKJCNXAC	Establish communication area, end an activation request, change settings specified by the user, and grant or deny CONSOLE command authority to a user.
	80% Message capacity IKJCNX50	Take action when the solicited or unsolicited message table becomes 80% full.
	100% Message capacity IKJCNX64	Take action when the solicited or unsolicited message table becomes 100% full.
	Deactivation IKJCNXDE	Perform clean-up processing.
CONSPROF command	Initialization IKJCNXCI	Check and, if necessary, change the command the user issues, update the profile with an installation portion, and grant or deny CONSOLE command authority to a user.
	Pre-display IKJCNXCD	Add information to the console profile display message IKJ55351I, or issue an installation-defined message instead of IKJ55351I.
	Termination IKJCNXCT	Perform clean-up processing, and store the installation portion of the console profile in a permanent place.
EDIT command	RENUM, MOVE, and COPY subcommands	Customize how the subcommands handle line numbering whenever a user issues the subcommands.
	Syntax checkers	Write an exit for syntax checkers that your installation provides. The exit fills in the option word with information the user specifies on the EDIT command.
EXEC command	Initialization IKJCT43I	Change the command the user issues.
	Termination IKJCT43T	Perform clean-up processing and set the defaults for the control characteristics of the CLISTs or REXX execs.
FREE command	Initialization IKJEFD21	Check and change the command users issue or provide pseudo-operands.
	Termination IKJEFD22	Perform clean-up processing.
Information Center Facility	ADRS	Add processing, such as displaying panels or allocating data sets, whenever a user selects the ADRS option from the Information Center Facility.
	Names service	Keep track of changes that are made to the private and master directories whenever Information Center Facility users use the names service.

Table 2. Overview of Exit Points that TSO/E Provides (continued)

Processing	Exit	Uses of Exit
LISTBC command	Initialization IKJEESX5	Initialize the environment for later exits, restrict users from using the command, or change the operands a user specifies on the command.
	Pre-display IKJEESX6	If using individual user logs, provide special formatting, append diagnostic information to a message, and support special features of output devices.
	Pre-list IKJEESX7	If using individual user logs, modify user log data set names and prepare for the pre-read exit.
	Pre-read IKJEESX8	If using individual user logs, tailor I/O.
	Pre-allocate IKJEESX9	If using individual user logs, allocate the user log data set.
	Failure IKJEESXA	If using individual user logs, perform failure processing and clean-up after an I/O failure.
	Termination IKJEESXB	Perform clean-up or special termination processing.
Logon and logoff processing	Logon pre-prompt IKJEFLD	Tailor the TSO/E logon process. Verify, change, or supply logon parameters and system characteristics, cancel logon requests, provide your own JCL statements, or display your own full-screen logon panel.
	Logon pre-prompt IKJEFLD1	Perform the functions of IKJEFLD, plus the following authorized functions: specify the first TSO/E command, return job and SYSOUT classes, bypass RACF, return the relative block address (RBA), pass data to the logoff exit, specify security label of current logon session, specify languages to be used for displaying translated information, and set up a console profile.
	Logon pre-display IKJEFLN1	Update information on the logon panel, process installation-defined fields on the logon panel.
	Logon post-display IKJEFLN2	Validate and process fields on the logon panel, re-prompt the user for information, and also request display of help screens.
	Logon post-prompt IKJEFLD3	Examine, modify, and add JCL statements associated with the logon process.
	Logoff IKJEFLD2	Tailor the TSO/E logoff process. Gather accounting information, control UADS and RACF data base updates, control re-logons.
OPERATOR SEND subcommand	See the entry for the SEND command in this table.	
OUTDES command	Initialization IKJEFY11	Check and change the command users issue or provide pseudo-operands.
	Termination IKJEFY12	Perform clean-up processing.

Table 2. Overview of Exit Points that TSO/E Provides (continued)

Processing	Exit	Uses of Exit
OUTPUT, STATUS, and CANCEL commands	IKJEFF53	Tailor the way users can handle the processing of batch jobs and their output.
PARMLIB command	Initialization IKJPRMX1	Change the command the user issues.
	Termination IKJPRMX2	Perform clean-up processing.
PRINTDS command	Initialization IKJEFY60	Tailor the fixed default values for specific operands. Restrict users from using the command or change the operands a user specifies on the command.
	Termination IKJEFY64	Perform clean-up processing.
RECEIVE	See the entry for the TRANSMIT command in this table.	
REXX processing	Pre-environment initialization IRXINITX	Perform processing after the initialization routine receives control but before the new language processor environment is initialized.
	Post-environment initialization IRXITTS or IRXITMV	Perform processing after the language processor environment is initialized but before the initialization routine returns control.
	Environment termination IRXTERMX	Perform processing before a language processor environment is terminated.
	Exec processing	Perform special processing before a REXX exec is executed.
	Exec initialization	Access and update REXX variables.
	Exec termination	Access and update REXX variables.
	Attention handling	Perform special attention processing in an environment integrated into TSO/E.
SEND command OPERATOR SEND subcommand	Initialization IKJEESX0 IEEVSNX0	Initialize the environment for later exits, change the defaults in SYS1.PARMLIB, restrict users from using the command, provide different user log data set names, and reroute messages by changing the user ID of the target user.
	Pre-display IKJEESX1 IEEVSNX1	If using individual user logs, provide special formatting, add diagnostic information, and support special features of output devices.
	Pre-save IKJEESX2 IEEVSNX2	If using individual user logs, override the user log data set name, support special I/O, and perform open and write operations. If used with the LISTBC pre-read exit, process the message or add information to it such as a sequence number, compress the message, and change parameters.
	Failure IKJEESX3 IEEVSNX3	If using individual user logs, perform failure processing and clean-up after an I/O failure.
	Termination IKJEESX4 IEEVSNX4	Perform clean-up or special termination processing.

Table 2. Overview of Exit Points that TSO/E Provides (continued)

Processing	Exit	Uses of Exit
Session Manager	Initialization	Indicate which streams you want to monitor and whether the Session Manager should log line mode output while users are executing full-screen programs.
	Stream monitoring	Monitor the individual streams you specified in the initialization exit and perform required processing.
	Termination	Perform special processing before the Session Manager ends.
STATUS	See the entry for the OUTPUT command in this table.	
SUBMIT command	IKJEFF10	Check submitted JCL statements and accept, reject, or modify them.
TEST command	Initialization IKJEGMIE	Change the command the user issues.
	Termination IKJEGMTE	Perform clean-up processing.
	Subcommand Initialization IKJEGCIE	Change the subcommand the user issues.
	Subcommand Termination IKJEGCTE	Perform clean-up processing.
TESTAUTH command	Initialization IKJEGAUI	Perform security verification.
	Termination IKJEGAUT	Perform clean-up processing.
	Subcommand Initialization IKJEGASI	Change the subcommand the user issues.
	Subcommand Termination IKJEGAST	Perform clean-up processing.

Table 2. Overview of Exit Points that TSO/E Provides (continued)

Processing	Exit	Uses of Exit
TRANSMIT and RECEIVE commands	**Common exit:** NAMES data set pre-allocation INMCZ21R **TRANSMIT exits:** Startup INMXZ01 INMXZ01R Log data set pre-allocation INMXZ21R Encryption INMXZ03 INMXZ03R Termination INMXZ02 INMXZ02R **RECEIVE exits:** Initialization INMRZ01 INMRZ01R Log data set pre-allocation INMRZ21R Data set pre-processing INMRZ11 INMRZ11R Data set decryption INMRZ13 INMRZ13R Notification NMRZ04 INMRZ04R Acknowledgement notification INMRZ05R Pre-acknowledgement notification INMRZ06R Data set post-processing INMRZ12 INMRZ12R Post-prompt INMRZ15R Termination INMRZ02 INMRZ02R	Use the various TRANSMIT and RECEIVE exits together to perform different types of processing. Use the exits to monitor transmission activity or customize how TRANSMIT and RECEIVE operate. For example, you can use the exits to: • Collect statistics about network usage • Control which users can use a particular network path • Support data sets that the Interactive Data Transmission Facility does not support • Tailor encryption and decryption processing • Suppress reception of data sets • Determine what action to take with an arriving transmission after a user response • Send an acknowledgment message back, using the receiver's security token.
TSOLIB command	Initialization IDYTSINI	Change the command the user issues.
	Termination IDYTSTER	Perform clean-up processing.

Chapter 5. Administration

Administration consists of managing the data processing resources of TSO/E and TSO/E users. The following sections describe the administrator's tasks and the facilities that TSO/E provides to perform the tasks. For complete information, see *z/OS TSO/E Administration*.

Administering TSO/E

You can define users to TSO/E and maintain user type definitions using the ENROLL option of the Information Center Facility, RACF commands, or the TSO/E ACCOUNT command. You can also use RACF commands and the TSO/E ACCOUNT command to remove users from TSO/E.

ENROLL option of the Information Center Facility

The Information Center Facility ENROLL option uses panels to gather information about a user you want to define to TSO/E. The ENROLL option relies on the USERTYPE option, which defines classes of users. Associated with each user type is the information necessary to define a user to TSO/E and define the type of access the user has to system resources. When you choose the ENROLL option, you supply unique information about the user including the user type.

Using the ENROLL option of the Information Center Facility has the following advantages:

- You can define users to both TSO/E and RACF at the same time, no matter what level of RACF is used at your installation.
- You can easily define users to the master catalog.
- You can include information about the user in the Information Center Facility names directory.
- You can easily add an ISPF/PDF user profile from the enrollment function.

Your installation can use security labels if RACF is active. To define security labels (SECLABELs) for users, you must use RACF commands. The Information Center Facility does not support security labels. See *z/OS Security Server RACF Security Administrator's Guide* for information about defining security labels for users.

RACF commands

If your installation has RACF active, you can use RACF commands to define users to both TSO/E and RACF, maintain those definitions in the RACF data base, and remove users. The RACF command:

- ADDUSER TSO adds a new user to TSO/E.
- ALTUSER TSO changes information about a previously-defined user.
- ALTUSER NOTSO removes a user from TSO/E.
- LISTUSER TSO lists information from the RACF data base about a user.

Using the RACF commands to add, maintain, and delete users has the advantage that (1) you can use one set of commands to add, change, or delete information about TSO/E user IDs and RACF security and (2) you need maintain only the RACF data base for user information. For more information about RACF commands, see *z/OS Security Server RACF Command Language Reference*. For information about adding users to the RACF data base, see *z/OS Security Server RACF Security Administrator's Guide*.

TSO/E ACCOUNT command

You can use the ACCOUNT command and its subcommands to define users to TSO/E, maintain those definitions in the user attributes data set (UADS), and remove users. If you have neither the Information Center Facility nor RACF, you must use the ACCOUNT command.

For information about the ACCOUNT command, see *z/OS TSO/E Administration*.

Converting to the RACF data base

If you have RACF and have information about users in both the UADS and the RACF data base, you can merge the information into the RACF data base with the RACONVRT utility. You can then maintain the one data base with RACF commands.

For information about the RACONVRT utility, see *z/OS TSO/E Customization*.

Administering the TSO/E Information Center Facility

The Information Center Facility contains two conversational, panel-driven interfaces. One enables users to access products and services and the other enables administrators to maintain the Information Center Facility. Each interface begins with a main menu panel that displays a list of options with descriptions and characters for selecting the options. If your installation has not customized the Information Center Facility, your main menu panel for administrator tasks will look something like the panel shown in Figure 4 on page 50.

```
              TSO/E Information Center Facility Administration
Option ===>

Select one of the following options.  To scroll, press UP or DOWN.

    0 DESCRIBE    - Read a short description of the options on this panel
    1 NEWS        - Maintain System News
    2 NAMES       - Maintain the name/phone directory
    3 ENROLL      - Enroll users in the Information Center Facility
    4 USERTYPE    - Set defaults for user types
    5 SYSDEF      - Set system defaults
    6 ICFUSER     - Use Information Center Facility user services
    7 COURSES     - Maintain education services
    8 PDF         - Use ISPF/PDF services
    9 PROBLEM     - Use problem reporting services
    I INTRO       - Learn to use the Information Center Facility
    T TUTORIAL    - Read detailed descriptions of options on this panel
    X EXIT        - Exit


To view PF key definitions, type KEYS on COMMAND or OPTION line of any panel.
```

Figure 4. Main Menu Panel For Administrator Tasks

In the figure above, options 1, 2, 3, 4, 5, and 7 describe the six major administrative tasks. When you select one of those options, the main menu leads to panels on which you can perform the specific task. The following sections describe the six major tasks that you would perform as an Information Center Facility administrator.

NEWS — Maintain System News

The Information Center Facility news service provides users with up-to-date news about courses, new products and other topics of interest. You are responsible for maintaining the news service. You can add, view, modify, and delete on-line news items. With TSO/E, you can also print news items.

NAMES — Maintain the Name/Phone Directory

The names directory contains entries for everyone enrolled in the Information Center Facility and possibly for other people as well. An entry can contain information about an individual or a group of people. You are responsible for maintaining the master copy of the names directory that all users can view. You can view, add, modify, and delete entries in the master names directory. When a user requests a change, you must view the request and decide whether to accept or reject it.

The TRANSMIT/RECEIVE Names File

The TSO/E TRANSMIT and RECEIVE commands use a names file (NAMES.TEXT) to send messages and data. This file uses information from the master names directory. If you make a change to the master

names directory, you must also update the NAMES.TEXT file using the appropriate option on the names directory menu panel before you exit the names directory option.

ENROLL — Enroll Users in the ICF

The ENROLL option allows you to enroll people through the Information Center Facility. The enrollment service is a simplified way of adding users to TSO/E and identifying them to the Information Center Facility names directory and, if installed, to RACF. You can also use the enrollment service to create a user catalog alias for the person's TSO/E user ID.

With RACF, your installation can use security labels. You must use RACF commands to define security labels (SECLABELs) for users. The Information Center Facility does not support security labels. See *z/OS Security Server RACF Security Administrator's Guide* for information about defining security labels for users.

USERTYPE — Set Defaults for User Types

The USERTYPE option allows you to maintain user type definitions. When enrolling a person through the Information Center Facility, you assign that person a user type. The user type identifies a class of users, such as administrator or user. Associated with each user type are the parameters the system uses when issuing the TSO/E ACCOUNT ADD, RACF ADDUSER TSO, and VSAM DEFINE ALIAS commands during enrollment processing. You can use Information Center Facility panels to add, modify, delete, or view user type definitions.

SYSDEF — Set System Defaults

The SYSDEF option allows you to set system defaults. The Information Center Facility has default definitions for users' ISPF profiles, for printer support, and for applications in the Information Center Facility.

ISPF Profile

The user's ISPF profile controls communication between the system and the user's terminal. The profile contains various information, such as terminal characteristics and PF key settings. During enrollment, the system copies the system default ISPF profile to initialize a profile for the person. You can then tailor the copied information to suit the person being enrolled. You can also change the system defaults when the standard defaults no longer apply to the average user being enrolled. Changing the system defaults requires you to do less customization when enrolling users.

For TSO/E, the system default ISPF profile is shipped as part of the Information Center Facility. You can use the SYSDEF option to modify the system default ISPF profile.

For TSO/E, the Information Center Facility uses the system default ISPF profile. You can use the SYSDEF option to modify the initial defaults supplied by ISPF/PDF. When you modify the initial defaults supplied by ISPF/PDF, a local copy of the profile is maintained by the Information Center Facility. In this case, the defaults supplied by ISPF/PDF are no longer used.

Printer Support

The printer support definition describes the printers available to your facility and their print characteristics. You can set up a print definition by entering information about print characteristics on Information Center Facility panels. You can also copy, modify, delete, and test an existing print definition.

Applications

The Information Center Facility is made up of different services that users can access. You can use the Application Manager to define services to the Information Center Facility. To be defined by Application Manager, a service must be broken down into parts called applications. There are three types of applications: environments, functions and panels. An environment defines the support for the service, such as the setup commands and the required libraries. A function is the CLIST, REXX exec, command,

program, or menu panel that invokes the service. A panel is the interface that allows users to select different services, tutorials, and HELP information.

With TSO/E, Application Manager supports three levels of application definitions: system, group, and private. This support allows departments, user groups, and individual users to create or tailor application definitions for their own use. Application definitions at the group or private level can be either unique definitions or copies of higher-level definitions. This support also enables installations to limit the availability of selected services to certain users. For example, the applications that define a service can be copied down from the system level to the group level for a specific group of users. The applications can then be deleted at the system level.

To administer system level application definitions, you invoke Application Manager using the SYSDEF option on the administration main menu panel. To administer group or private application definitions, you invoke Application Manager using the PROGRAM option on the end user main menu panel.

For information about accessing application definitions defined for a specific group, see "Group Specification" on page 11.

Application Manager simplifies the task of integrating products and functions into the Information Center Facility. You can add a service by either loading installation files that define the service in terms of its applications or by entering specific information on Application Manager panels. After you add a service, you can copy its applications, modify them, view them, delete them, and determine where they are used. You can also replace an existing application with an application of the same name, language, and application type.

You can use the EXPORT option to create an installation file from an existing application. The EXPORT option makes it possible to move Application Manager definitions installed on one system to other systems without having to repeat the initial installation process.

You can specify alternate application libraries for CLISTs and REXX execs. The Application Manager invocation processing uses the ALTLIB command to dynamically allocate these libraries when a user invokes an application. The ALTLIB command eliminates the need for allocating user and application CLIST and REXX exec libraries to a fixed ddname in the user's logon procedure.

With TSO/E, you can also:

- Use the HIERARCHY option to display a hierarchy of panels and functions for any defined panel application.
- Use the upgrade function to upgrade an existing installation file with the contents of an upgrade file. An upgrade file is in the format of an installation file and contains entries for adding, deleting, and replacing portions of an installation file. Using upgrade files to apply changes to installation files minimizes the possibility that the changes will conflict with any customization that may have been done.
- Specify any library required by an application. To differentiate the libraries specified, a library TYPE field is included on the Application Libraries panel. The valid library types are (1) ISPF for ISPF files, (2) CLIST for CLIST files, (3) EXEC for REXX files, (4) INPUT for any other file type with the intended use of input during invocation, (5) OUTPUT for any other file type with the intended use of output during invocation.

All Application Manager menu panels have the same format. In addition, the menu panels are scrollable so that installations can fit more options on them.

With TSO/E, you can process multiple installation files with or without upgrade files and export a list of applications without the use of Application Manager panels. The mass installation file process provides two separate functions. These are:

- Mass upgrade and install

  This function uses upgrade and/or installation files to upgrade or replace existing applications and install new applications. Each installed application is verified and made available.

  You can mass process installation files with or without corresponding upgrade files. You can also mass process upgrade files with or without corresponding installation files.
- Mass export

This function uses a list of applications to drive an automated export process. Applications are exported to the output installation library

You can invoke the mass installation file process in the batch TMP (no active terminal) or in the foreground (active terminal).

Installations can use several exits to tailor the processing that occurs when users invoke applications defined by Application Manager. For a description of the exits, see *z/OS TSO/E Customization*.

COURSES — Maintain Education Services

The COURSES option allows you to maintain the education services. Education Services give users access to computer and classroom courses and courses your installation offers using the Interactive Instructional Presentation System (IIPS) and the Interactive Instructional Authoring System (IIAS). You are responsible for maintaining on-line course information. You can add, modify, and delete courses and course abstracts. With TSO/E, you can also print course abstracts. You are also responsible for processing registration requests from users and maintaining course registration lists, so users know in which courses they are registered.

Chapter 6. Diagnosis

Diagnosis involves reading system messages and reading and analyzing dumps to determine where a problem occurred and, if necessary, re-creating the problem with tracing facilities in effect to obtain further information. The final step, if necessary, is to report the problem to your IBM support center and supply the necessary information. The following topics describe the tools that TSO/E provides to help you diagnose errors.

Messages and codes

You can identify the issuing component by the first three letters in the message identifier (message ID) that precedes the message. You can find explanations of the messages, and possible actions to take in response, in *z/OS TSO/E Messages*. You can also obtain online help for some terminal messages by issuing the HELP command with the message ID as shown in the following example:

```
HELP command name MSGID (message ID)
```

Error messages are sometimes accompanied by abend and reason codes. These codes also appear in the summary sections of an ABEND dump. An abend is an abnormal termination of a component. Sxxx identifies a system abend code, caused by a system error, and Uxxxx identifies a user abend code, caused by a user's program or action. The four-digit reason code follows the abend code. For an explanation of these codes, see *z/OS MVS System Codes*.

Tracing execution of programs and installation exits

TSO/E provides the following tools to help you trace the execution of a program or installation exit and obtain more information about an error condition:

- Trace function of the Information Center Facility
- TEST and TESTAUTH commands
- Debug facilities for REXX Execs
- OPERATOR SLIP command support for tracing installation exits

TRACE function of the Information Center Facility

The Information Center Facility is written in the TSO/E CLIST and REXX languages.

The Information Center Facility's TRACE function allows you to trace the execution of the facility's CLISTs/REXX execs. There are three levels of tracing available. Level 1 shows the order in which nested CLISTs/REXX execs are invoked and executed. Level 2 provides the same trace as level 1 and also displays detailed information for each nested CLIST/REXX exec. For each nested CLIST, it displays the CLIST statements, TSO/E command and subcommands, and informational messages from the CLIST, before execution. Each of the nested CLISTs executes with the following diagnostic options in effect:

```
CONTROL SYMLIST CONLIST LIST MSG
```

For more information about these options, see *z/OS TSO/E CLISTs*.

For each nested REXX exec, level 2 tracing displays each clause before execution. Clauses include null clauses, labels, assignments, keyword instructions, and commands. Each of the nested REXX execs executes with the following diagnostic option in effect:

```
TRACE INTERMEDIATES
```

For more information about this option, see *z/OS TSO/E REXX Reference*.

For all Information Center Facility CLISTs/REXX execs, level 3 provides the same trace as level 1. In addition, for a single explicitly-named CLIST/REXX exec, level 3 tracing provides the same trace as level 2.

For each level of tracing, the Information Center Facility displays the name of the CLIST/REXX exec when it is invoked and when it terminates, with the final condition code from the CLIST/REXX exec displayed at termination. For more information about the trace function of the Information Center Facility, see *z/OS TSO/E Customization*.

TSO/E TEST and TESTAUTH commands

The TSO/E TEST and TESTAUTH commands enable you to diagnose problems in assembler language programs. You can use the TEST command to test unauthorized programs including those that use access registers. You can use the TESTAUTH command to test APF-authorized programs. You must be defined to the RACF TESTAUTH resource to use TESTAUTH.

While you are executing a program that you have written, you might encounter errors or abnormal terminations (ABENDs). You can use the TEST command to help determine the cause of errors in a program that is currently executing. You can also use the TEST command to load and execute a program, and monitor the program's execution.

You can use the TESTAUTH command to test an APF-authorized program that is not currently executing. Unlike the TEST command, you cannot use TESTAUTH to test a currently executing program.

The TEST and TESTAUTH commands are similar in that they support most of the same operands, subcommands, and functions. After you enter test mode, you can issue the subcommands of TEST or TESTAUTH to perform various functions such as:

- Stepping through sections of the program to check each instruction for proper execution
- Interrupting the program at specified locations
- Viewing storage contents
- Changing the contents of specified program locations in virtual storage or the contents of specific registers

For example, you can use the LIST subcommand to display:

- The contents of a specified area of storage
- The contents of registers or vector registers
- The contents of access registers
- Data in alternate address/data spaces that is referred to using an access register
- The vector mask register

With TSO/E, you can also display the partial sum number and the vector section size of a vector machine.

With TSO/E, you can use the TEST command to test load modules maintained in extended partitioned data sets (PDSEs).

For more information about using the TEST and TESTAUTH commands to test programs and locate programming errors, see *z/OS TSO/E Programming Guide*.

Debug facilities for REXX execs

TSO/E provides several facilities to help you locate errors in REXX execs. These are:

- The REXX TRACE instruction and built-in function
- The interactive debug facility
- TSO/E REXX immediate commands and the TSO/E EXECUTIL command
- The IRXIC trace and execution control routine

For complete information about the REXX debug facilities, see *z/OS TSO/E REXX Reference*.

REXX TRACE instruction and built-in function

You can use the REXX TRACE instruction or the REXX TRACE built-in function in a REXX exec to display how the language processor evaluates each operation in the exec. By specifying options on the instruction or built-in function, you can control the tracing action taken (how much will be displayed to the user at the terminal) during execution of the exec. The TRACE built-in function also lets you display the trace actions currently in effect.

Using the prefix option **?** on the TRACE instruction or TRACE built-in function, you can control the interactive debug facility. During normal execution, a TRACE instruction or built-in function prefixed with ? will turn on interactive debug. Any additional TRACE instructions in the exec being traced are ignored. The TRACE built-in function, however, can alter the trace action even if interactive debug is active.

Interactive debug facility

The interactive debug facility permits interactively controlled execution of an exec. You can interactively debug REXX execs in the TSO/E address space from your terminal session. The language processor pauses after most instructions that are traced at the terminal to let the user specify the debug action to be taken. You can specify that the language processor continue execution until the next pause for debug input, re-execute the clause last traced, or immediately process the instructions that you enter. Because any REXX instructions may be executed in interactive debug, you have considerable control over execution.

TSO/E REXX immediate commands and the EXECUTIL command

You can use four TSO/E REXX immediate commands (HI, HE, TS, TE) and the TSO/E EXECUTIL command with three operands (HI, TS, and TE) to interrupt execution and control tracing of a REXX exec that executes in the TSO/E address space. The TS and TE immediate commands can also be used to control the tracing of a REXX exec that executes in a non-TSO/E address space.

- The HI immediate command or the EXECUTIL HI command halts the interpretation of all REXX execs currently executing in the language processor environment. The execs are terminated after control returns to the currently running exec. These commands are useful if a REXX exec gets into a loop and you want to halt interpretation of the exec.
- The HE immediate command halts the execution of the currently executing REXX exec. The exec is terminated before control is returned to the currently executing exec. This command is useful to terminate a looping function, subroutine, or host command not written in REXX that has been invoked by a REXX exec.

  IBM recommends that you use HI instead of HE whenever possible. HE will not halt execution of any exec that called the currently running exec unless you are running under ISPF.

  Note: HE is not a valid operand of the EXECUTIL command.
- The TS immediate command or the EXECUTIL TS command puts the REXX exec into normal interactive debug and you can then execute REXX instructions as you would normally. These commands are also useful if you suspect a REXX exec is looping. You can step through the exec before you decide whether to allow the exec to continue or not.
- The TE immediate command or the EXECUTIL TE command stops tracing. These commands can be useful to stop tracing when not in interactive debug.

In the TSO/E address space, you can use the HI, HE, TS, and TE immediate commands by pressing the attention key to enter attention mode and then entering the command. In any address space, you can use the TS and TE immediate commands in an exec.

You can use the EXECUTIL command with the HI, TS, and TE operands:

- In an exec that executes in the TSO/E address space.
- From TSO/E READY mode and ISPF.
- In a TSO/E CLIST.
- In a program written in a high-level programming language by using the TSO/E service facility.

Trace and execution control routine (IRXIC)

TSO/E also provides a trace and execution control routine (IRXIC) which is an interface to the REXX immediate commands HI, HT, RT, TE, and TS. A program, in any MVS address space, can call IRXIC to use one of these commands to control the tracing and execution of REXX execs.

Using the OPERATOR SLIP command to diagnose installation exits

When diagnosing an installation exit, you can use the OPERATOR SLIP command to trace the exit points on the call and return from the exit and display the exit parameter list. IKJCALLX is the label of the tracing point that is executed before control is given to an installation exit. IKJRETNX is the label of the tracing point that is executed after control is returned from the exit.

TSODATA dump formatter

The TSODATA dump formatter allows you to format the following information from system dumps to solve problems related to TSO/E:

- TSO/E control blocks, such as CNCCB, ECT, LWA, and LSD
- CLIST symbol variables
- I/O services stack elements
- REXX control blocks.

On any z/OS system, you can use the TSODATA dump formatter in the interactive problem control system (IPCS). You invoke TSODATA from IPCS using the VERBEXIT command. With IPCS, you can use line-mode TSO/E or an IPCS/ISPF dialog management full-screen facility to examine formatted storage dumps on-line. IPCS also provides a facility for analysis of key TSO/E system components and control blocks. For more information about invoking TSODATA from IPCS, see *z/OS MVS IPCS Commands*.

TSO/E health checks

TSO/E provides several health checks that are registered and activated automatically at IPL. Among other things, these health checks alert operators whether there is a TSO/E configuration problem due to an error during system initialization. They also check whether user logs have been implemented in order to reduce dependence on the system broadcast data set. System settings can be changed or refreshed in response to a warning from the health checks to avoid or correct a problem. Checks can also be disabled with MVS or SDSF commands. The goal of the checks is to detect potential problems early and enhance reliability. See *IBM Health Checker for z/OS User's Guide* for more information about the Health Checker facility.

Appendix A. Accessibility

Accessible publications for this product are offered through IBM Documentation (www.ibm.com/docs/en/zos).

If you experience difficulty with the accessibility of any z/OS information, send a detailed message to the Contact the z/OS team web page (www.ibm.com/systems/campaignmail/z/zos/contact_z) or use the following mailing address.

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
United States

Accessibility features

Accessibility features help users who have physical disabilities such as restricted mobility or limited vision use software products successfully. The accessibility features in z/OS can help users do the following tasks:

- Run assistive technology such as screen readers and screen magnifier software.
- Operate specific or equivalent features by using the keyboard.
- Customize display attributes such as color, contrast, and font size.

Consult assistive technologies

Assistive technology products such as screen readers function with the user interfaces found in z/OS. Consult the product information for the specific assistive technology product that is used to access z/OS interfaces.

Keyboard navigation of the user interface

You can access z/OS user interfaces with TSO/E or ISPF. The following information describes how to use TSO/E and ISPF, including the use of keyboard shortcuts and function keys (PF keys). Each guide includes the default settings for the PF keys.

- *z/OS TSO/E Primer*
- *z/OS TSO/E User's Guide*
- *z/OS ISPF User's Guide Vol I*

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users who access IBM Documentation with a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line because they are considered a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that the screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1)

are mutually exclusive alternatives. If you hear the lines `3.1 USERID` and `3.1 SYSTEMID`, your syntax can include either `USERID` or `SYSTEMID`, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The `*` symbol is placed next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element `*FILE` with dotted decimal number 3 is given the format `3 \* FILE`. Format `3* FILE` indicates that syntax element `FILE` repeats. Format `3* \* FILE` indicates that syntax element `* FILE` repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol to provide information about the syntax elements. For example, the lines `5.1*, 5.1 LASTRUN, and 5.1 DELETE` mean that if you use more than one of the `LASTRUN` and `DELETE` syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, it indicates a reference that is defined elsewhere. The string that follows the % symbol is the name of a syntax fragment rather than a literal. For example, the line `2.1 %OP1` means that you must refer to separate syntax fragment `OP1`.

The following symbols are used next to the dotted decimal numbers.

? indicates an optional syntax element
The question mark (?) symbol indicates an optional syntax element. A dotted decimal number followed by the question mark symbol (?) indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example `5? NOTIFY`). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines `5 ?, 5 NOTIFY`, and `5 UPDATE`, you know that the syntax elements `NOTIFY` and `UPDATE` are optional. That is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

! indicates a default syntax element
The exclamation mark (!) symbol indicates a default syntax element. A dotted decimal number followed by the `!` symbol and a syntax element indicate that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the dotted decimal number can specify the ! symbol. For example, if you hear the lines `2? FILE, 2.1! (KEEP), and 2.1 (DELETE)`, you know that `(KEEP)` is the default option for the `FILE` keyword. In the example, if you include the `FILE` keyword, but do not specify an option, the default option KEEP is applied. A default option also applies to the next higher dotted decimal number. In this example, if the `FILE` keyword is omitted, the default `FILE(KEEP)` is used. However, if you hear the lines `2? FILE, 2.1, 2.1.1! (KEEP)`, and `2.1.1 (DELETE)`, the default option KEEP applies only to the next higher dotted decimal number, `2.1` (which does not have an associated keyword), and does not apply to `2? FILE`. Nothing is used if the keyword `FILE` is omitted.

*** indicates an optional syntax element that is repeatable**
The asterisk or glyph (*) symbol indicates a syntax element that can be repeated zero or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line `5.1* data area`, you know that you can include one data area, more than one data area, or no data area. If you hear the lines `3* , 3 HOST, 3 STATE`, you know that you can include `HOST, STATE`, both together, or nothing.

Notes:

1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you can write `HOST STATE`, but you cannot write `HOST HOST`.
3. The * symbol is equivalent to a loopback line in a railroad syntax diagram.

+ indicates a syntax element that must be included

The plus (+) symbol indicates a syntax element that must be included at least once. A dotted decimal number followed by the + symbol indicates that the syntax element must be included one or more times. That is, it must be included at least once and can be repeated. For example, if you hear the line `6.1+ data area`, you must include at least one data area. If you hear the lines `2+, 2 HOST,` and `2 STATE`, you know that you must include `HOST, STATE,` or both. Similar to the * symbol, the + symbol can repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loopback line in a railroad syntax diagram.

Notices

This information was developed for products and services that are offered in the USA or elsewhere.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to 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 information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

This information could include missing, incorrect, or broken hyperlinks. Hyperlinks are maintained in only the HTML plug-in output for IBM Documentation. Use of hyperlinks in other output formats of this information is at your own risk.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Site Counsel
2455 South Road

Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or

reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's name, email address, phone number, or other personally identifiable information for purposes of enhanced user usability and single sign-on configuration. These cookies can be disabled, but disabling them will also eliminate the functionality they enable.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at ibm.com®/privacy and IBM's Online Privacy Statement at ibm.com/privacy/details in the section entitled "Cookies, Web Beacons and Other Technologies," and the "IBM Software Products and Software-as-a-Service Privacy Statement" at ibm.com/software/info/product-privacy.

Policy for unsupported hardware

Various z/OS elements, such as DFSMSdfp, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those

products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: IBM Lifecycle Support for z/OS (www.ibm.com/software/support/systemsz/lifecycle)
- For information about currently-supported IBM hardware, contact your IBM representative.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at Copyright and Trademark information (www.ibm.com/legal/copytrade.shtml).

Index

Special Characters

<TSO/E>
    summary of changes xiii

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

Product Number: 5650-ZOS

SA32-0979-50